

AD-A073 976

NAVAL POSTGRADUATE SCHOOL MONTEREY CA

F/6 15/7

PARAMETRIC TERRAIN AND LINE OF SIGHT MODELLING IN THE STAR COMB--ETC(U)

AUG 79 J K HARTMAN

MIPR-CD-1-79

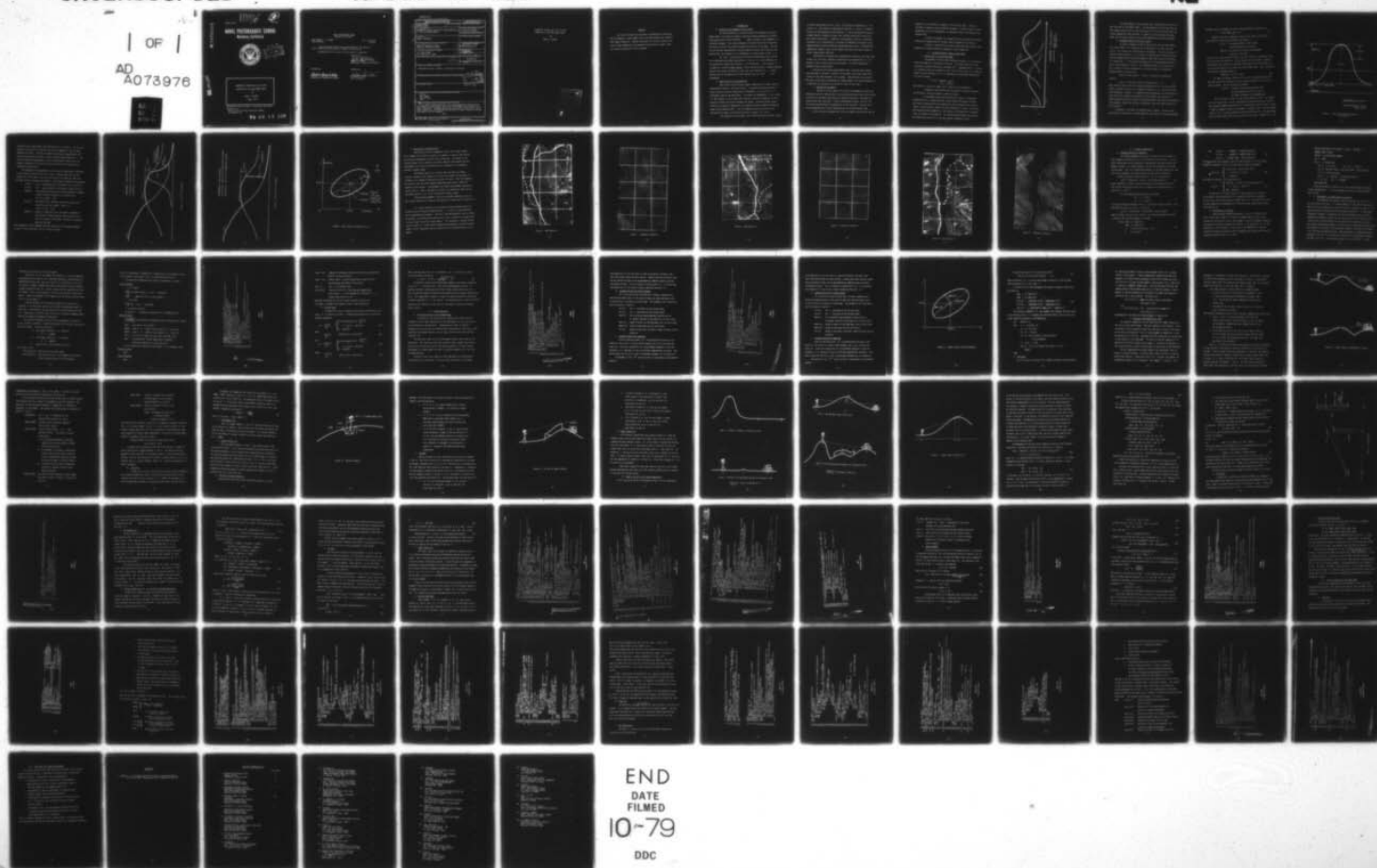
UNCLASSIFIED

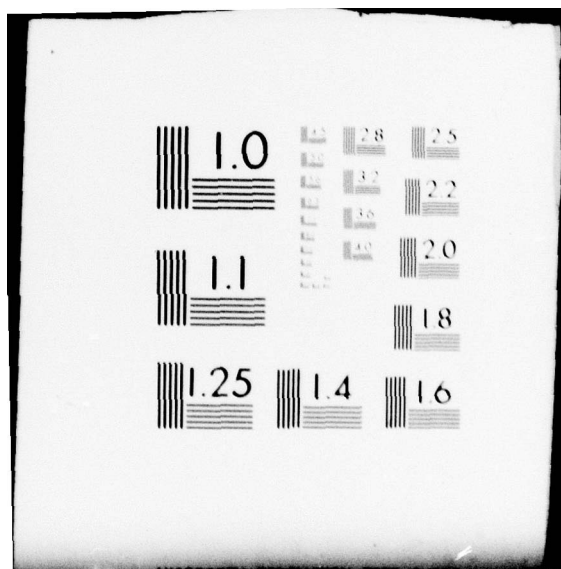
NPS55-79-018

NL

| OF |

AD
A073976





AD A 073976

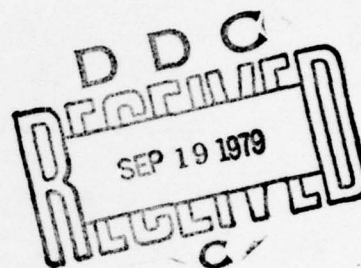
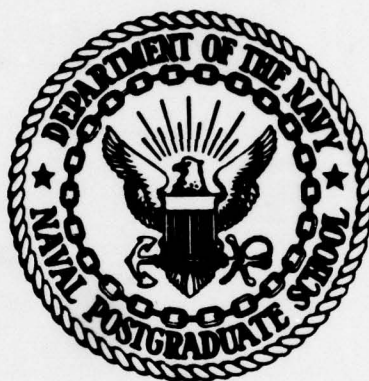
LEVEL



NPS55-79-018

NAVAL POSTGRADUATE SCHOOL

Monterey, California



PARAMETRIC TERRAIN AND LINE OF SIGHT
MODELLING IN THE STAR COMBAT MODEL

by

James K. Hartman

August 1979

Approved for public release; distribution unlimited.

Prepared for:

The U.S. Army Training & Doctrine Command
Fort Monroe, VA.

79 09 18 138

DDC FILE COPY

NAVAL POSTGRADUATE SCHOOL
Monterey, California

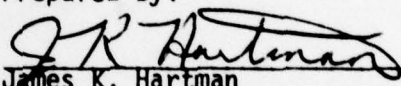
Rear Admiral T. F. Dedman
Superintendent

Jack R. Borsting
Provost

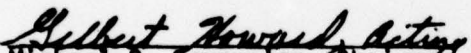
The work reported herein was accomplished with the support of
the U.S. Army Training & Doctrine Command, Fort Monroe, VA.

Reproduction of all or part of this report is authorized.

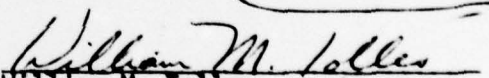
Prepared by:


James K. Hartman
Department of Operations Research

Reviewed by:


Michael G. Sovereign, Chairman
Department of Operations Research

Released by:


William M. Tolles
Dean of Research

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 14 NPS55-79-018 ✓	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) 6 Parametric Terrain and Line of Sight Modelling in the STAR Combat Model.	5. TYPE OF REPORT & PERIOD COVERED 9 Technical Report	
7. AUTHOR(s) 10 James K./Hartman	6. PERFORMING ORG. REPORT NUMBER	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School ✓ Monterey, California 93940	8. CONTRACT OR GRANT NUMBER(s)	
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army TRADOC Fort Monroe, VA 23651	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 15 MIPR ★ CD-1-79	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	12. REPORT DATE 11 August 1979	
12 88p.	13. NUMBER OF PAGES 85	
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.	15. SECURITY CLASS. (of this report) UNCLASSIFIED	
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Terrain Land Combat Line of Sight STAR		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report presents the motivation and mathematical background for the parametric terrain model used in the STAR Brigade level combined arms combat simulation. Computer subroutines for terrain elevation and line of sight computations are presented and explained in detail along with several preprocessor utility programs.		

DDC
RECEIVED
SEP 19 1979
C

DD FORM 1473

1 JAN 73

EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

251 450

JCB

Parametric Terrain and Line of Sight
Modelling in the STAR Combat Model

by

James K. Hartman

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DDC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	<input type="checkbox"/>
By _____	
Distribution/ _____	
Availability Codes	
Dist.	Avail and/or special
A	

ABSTRACT

This report presents the motivation and mathematical background for the parametric terrain model used in the STAR Brigade level combined arms combat simulation. Computer subroutines for terrain elevation and line of sight computations are presented and explained in detail along with several preprocessor utility programs.

I. INTRODUCTION

A. Origins of the Parametric Terrain Model

The STAR (Simulation of Tactical Alternative Responses) ground-air combat model is an extensive computer simulation program developed at the Naval Postgraduate School during 1978-1979. STAR is written in the SIMSCRIPT II.5 simulation language. This report documents the battlefield terrain representation which is used in the current Brigade-level version of the model. The idea for this terrain representation--called parametric terrain--was originally proposed by Maj. Chris Needels in his 1976 Master of Science thesis at NPS [1].

The basic function which any terrain representation must provide for a high-resolution combat simulation is, "for any X, Y map coordinates on the battlefield, compute the elevation Z of the terrain." This elevation Z is generally called the macro terrain. Macro terrain provides a somewhat smoothed replica of an actual battlefield in that very small features (e.g. a 1 meter boulder) are not represented, but major features (such as a 200 m hill) are represented.

B. Approaches to Terrain Modelling

Most current high resolution combat simulations use a macro terrain representation known as "digitized terrain". Essentially digitized terrain involves storing a (usually large) table of elevations, Z, for a grid of X, Y coordinates covering the battlefield. The process of determining Z for a given X,Y then reduces to one or more table look-ups, possibly followed by an interpolation to smooth the terrain between grid points. Digitized terrain enjoys simplicity and speed of computation, but requires extensive computer storage devoted to the elevation table. This has tended to restrict the size of the battlefield for models using digitized terrain to about 10 x 10 km.

The parametric terrain model used in STAR essentially involves storing

a (rather complicated) function $f(X,Y)$. The process of determining Z for a given X,Y then reduces to computing the function $Z = f(X,Y)$. Parametric terrain has the advantage that the function f can be represented using only a modest amount of computer storage, thus allowing simulations to be run on much larger battlefields. In addition, the parametric terrain is inherently continuous, so no interpolation is required for smoothing, and slopes can be computed directly with no difference approximations required. The associated computations, however, may be quite complex, and great care must be taken to achieve computational efficiency.

Since STAR is a Brigade level combined-arms simulation, with long range systems (air, artillery, resupply) represented on the battlefield, it is important to have a large terrain area available. Thus STAR incorporates a parametric macro terrain model.

Discussions with experienced modelers have indicated that attempts have been made in the past to develop terrain models which share some of the features of the STAR parametric terrain model. These efforts do not seem to have been successfully incorporated into combat models, but since documentation is generally not available, the reasons for this are not clear.

C. Overview of the Report

Section II of this report will develop the parametric terrain representation and will indicate how it can be used to emulate actual terrain areas. In Section III we detail the process of computing elevations, including the STAR routines ELEV and ELEVG. A terrain preprocessor program HILL.LIST for increasing the efficiency of computations is also discussed. Section IV discusses augmenting the terrain representation to include forested areas.

Given a terrain representation, one vital combat function which must be

computed is the existence or absence of line-of-sight (LOS). Section V discusses in general terms the STAR LOS procedure. Section VI goes into the mathematics of the LOS computations in substantial detail, and analyzes the computer code.

Section VII presents the computer code for the terrain and forest preprocessor programs and for the data read-in program for the terrain.

Finally, in Section VIII we discuss some areas which seem to have potential for further research in parametric terrain modelling.

II. The STAR Parametric Macro Terrain Model

A. General Form of the Terrain Model

The parametric terrain model proposed by Needels in [1] represents terrain by modelling individual hill masses. The overall terrain is then obtained by superpositioning the individual hills. Mathematically, if $f_I(X,Y)$ is a function giving the elevation of the I th hill mass at any X,Y , then the overall terrain elevation at X,Y is obtained as the pointwise maximum over all the hill masses,

$$Z = f(X,Y) = \text{maximum}_{I=1,2,\dots,NHILLS} f_I(X,Y) \quad (1)$$

where $NHILLS$ is the total number of hill masses on the battlefield.

A schematic cross-section view of several hill masses along with the resulting terrain elevation is shown in Figure 1.

In the Needels model each individual hill mass is represented mathematically as a scaled bivariate normal probability density function. This gives a characteristic bell-shaped hill mass cross-section as shown in Figure 1, and elliptical contours for each individual hill mass. By varying the bivariate normal parameters, a wide variety of different hill locations, sizes, and shapes can be modelled. By superpositioning several hill masses, the contour map can be fit to real map contours remarkably closely.

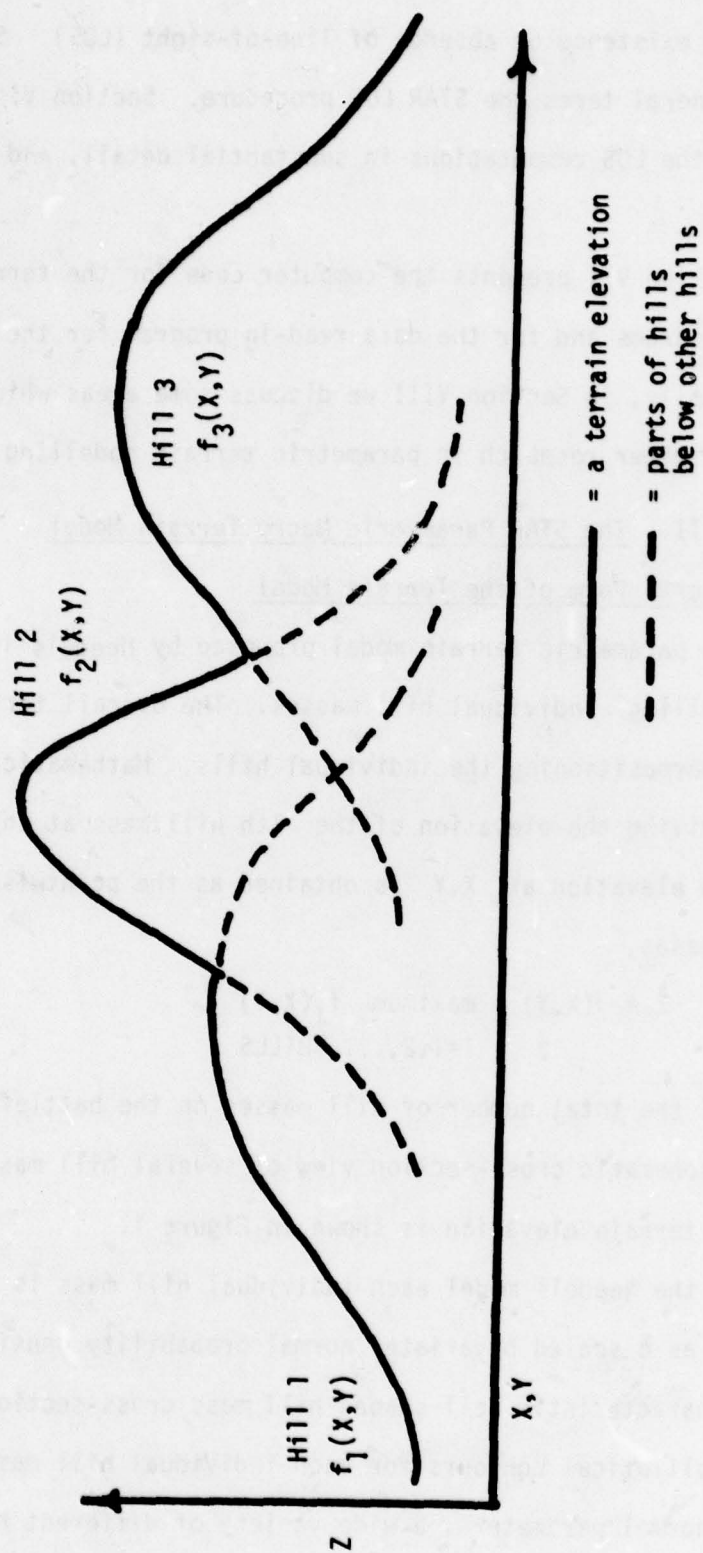


Figure 1. $f(X, Y) = \max_I f_I(X, Y)$

The STAR parametric terrain model uses a generalization of the hill mass functions of the Needels model. The representation of these functions has been changed significantly to make them easier to fit to real terrain maps and to make computations more efficient. In addition, forest features have been added, and the rather complex line-of-sight computations have been developed.

B. Parameterization for Terrain Fitting

In order to emulate a piece of real terrain with the STAR parametric terrain model, it is necessary to fit the model's hill mass functions $f_I(X,Y)$ to a contour map of the terrain to be modeled. The fitting process is currently done by hand, so it is essential to describe the hill-mass functions using a parameterization that makes geometric sense and can easily be related to a contour map. In this section we develop the "fitting-parameterization" of the hill mass functions $f_I(X,Y)$. This parameterization is not particularly well suited for efficient computation, so we will convert to a computing-parameterization in Section III.

Each hill mass function $f_I(X,Y)$ is given by an exponential function

$$f_I(X,Y) = \text{PEAK.H}(I) + \text{HT.H}(I) * [\exp(Q_I(X,Y)) - 1] \quad (2)$$

where $\text{PEAK.H}(I)$ and $\text{HT.H}(I)$ are constant parameters for hill I to be described shortly, and where $Q_I(X,Y)$ is a quadratic function of X,Y which has several other parameters. $Q_I(X,Y)$ is a negative definite quadratic, so the $\exp(Q_I(X,Y))$ term yields hills having the characteristic bell-shaped cross-section of a normal probability density function, and having contour lines which are elliptical. The parameters control the location, orientation, size, and shape of the elliptical contours.

Elevations in the model are measured from 0 meters = sea-level, and the terrain as a whole has a minimum elevation level above sea level denoted BASE.

The BASE value is included as a term in the maximization of equation (1)

$$Z = \max \{ \text{BASE}, \max_I f_I(X, Y) \} \quad (3)$$

Several of the hill parameters relate to hill height, and are best visualized on a cross-section diagram (see Figure 2.).

HT.H(I): the maximum height of the "normal" curve describing this hill mass.

PEAK.H(I): the elevation of the hilltop measured from zero = sea-level.

Since HT.H(I) may be greater than or less than PEAK.H(I), the bottom of the hill which occurs at PEAK.H(I) - HT.H(I) need not be at sea-level. However, since normal curves have tails approaching the bottom level which extend to $\pm \infty$ we require that these tails be below the terrain BASE value. Thus a constraint on the terrain parameterization is

$$\text{PEAK.H(I)} - \text{HT.H(I)} < \text{BASE} \quad (4)$$

For some hill masses, the tails of the hill are annoying as they prevent the terrain from dropping off quickly enough to model a cliff side or a steep riverbank. Thus we include the parameter

CUT.H(I): A vertical distance measured down from the peak beyond which this hill mass is no longer considered in the computations.

Thus if CUT.H(I) is 70m., at most seven contours of hill I will appear on a 10m. contour map. The CUT.H parameter must be used with care, since it may introduce unwanted discontinuities in the terrain--vertical cliffs. Figure 3 shows an example in which this parameter is properly used. Here the gradual slope of hill 2 has been terminated using CUT.H(2) and hill 3 has been added to give a more abrupt slope to the terrain at this point. Without hill 3 the

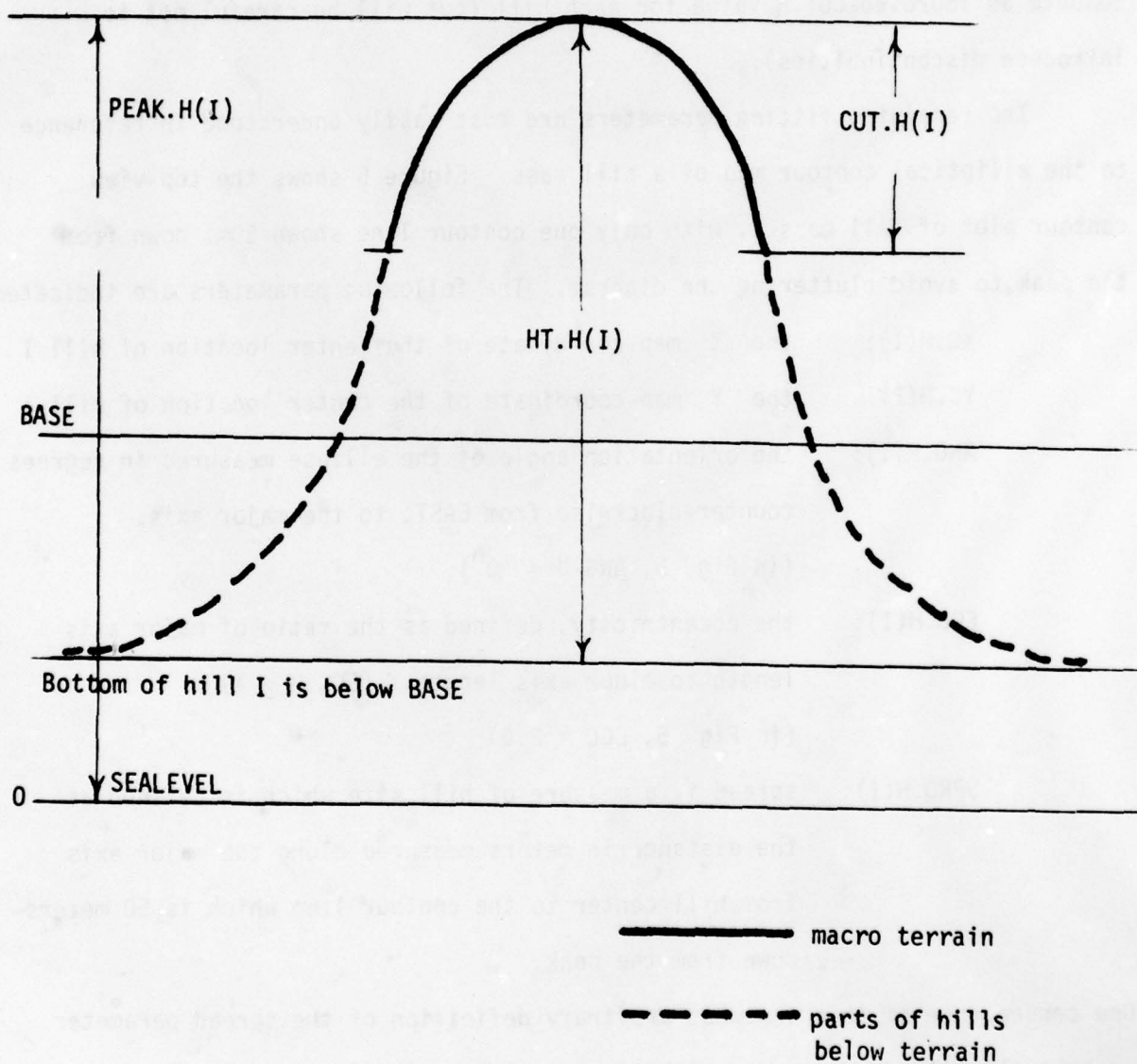


Figure 2. Cross-section through center of hill mass I.

resulting terrain would have a bad discontinuity as in Figure 4. For the vast majority of the hills not requiring the CUT.H(I) parameter it may be safely defaulted to HT.H(I). The CUT.H parameter also improves computational efficiency by restricting the region in which we need to worry about hill I. The Terrain Preprocessor program HILL.LIST to be described in Section III will compute an improved CUT.H value for each hill (but will be careful not to introduce discontinuities).

The remaining fitting parameters are most easily understood in reference to the elliptical contour map of a hill mass. Figure 5 shows the top view contour plot of hill mass I, with only one contour line shown 50m. down from the peak, to avoid cluttering the diagram. The following parameters are indicated:

- XC.H(I): the X map-coordinate of the center location of hill I
- YC.H(I): the Y map-coordinate of the center location of hill I
- ANG.H(I): the orientation angle of the ellipse measured in degrees counter-clockwise from EAST. to the major axis.
(in Fig. 5, ANG.H = 30°)
- ECC.H(I): the eccentricity, defined as the ratio of major axis length to minor axis length. ($ECC.H \geq 1$)
(in Fig. 5, ECC = 2.0)
- SPRD.H(I): spread is a measure of hill size which is defined as the distance in meters measured along the major axis from hill center to the contour line which is 50 meters down from the peak.

One consequence of this somewhat arbitrary definition of the spread parameter is that all hills must have a HT.H(I) greater than 50m.

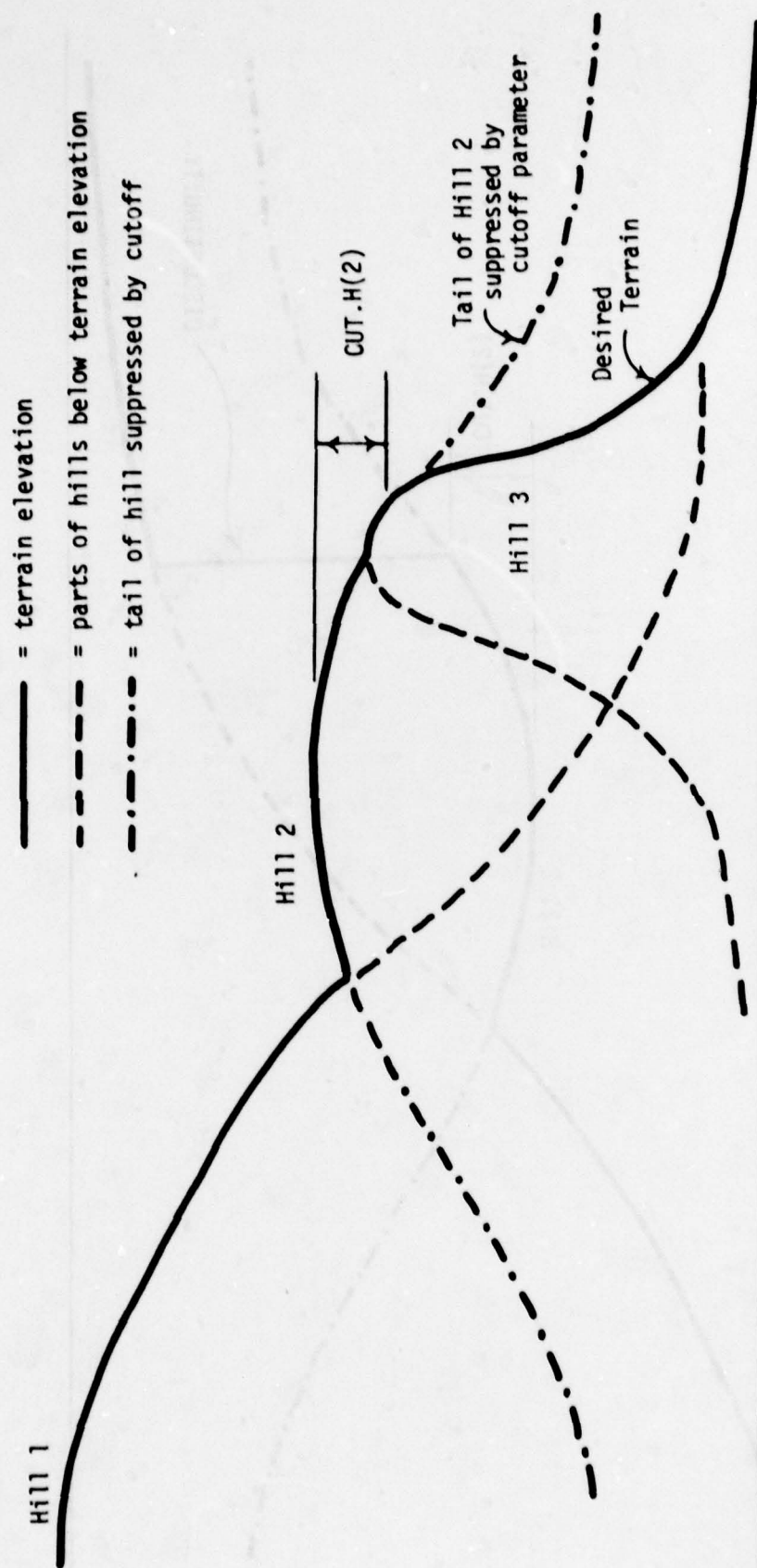


Figure 3. Use of CUT.H parameter

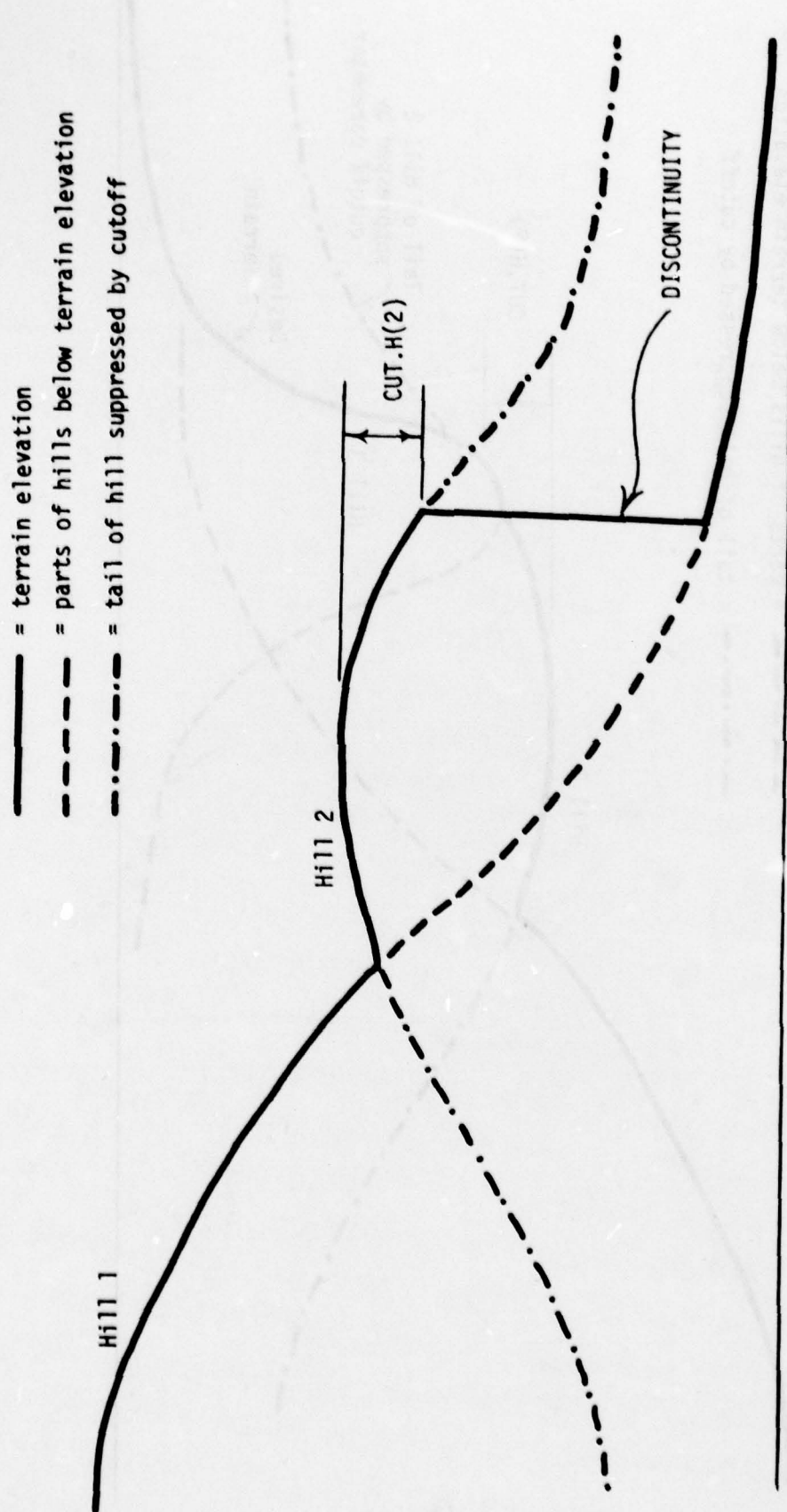


Figure 4. Misuse of CUT.H parameter

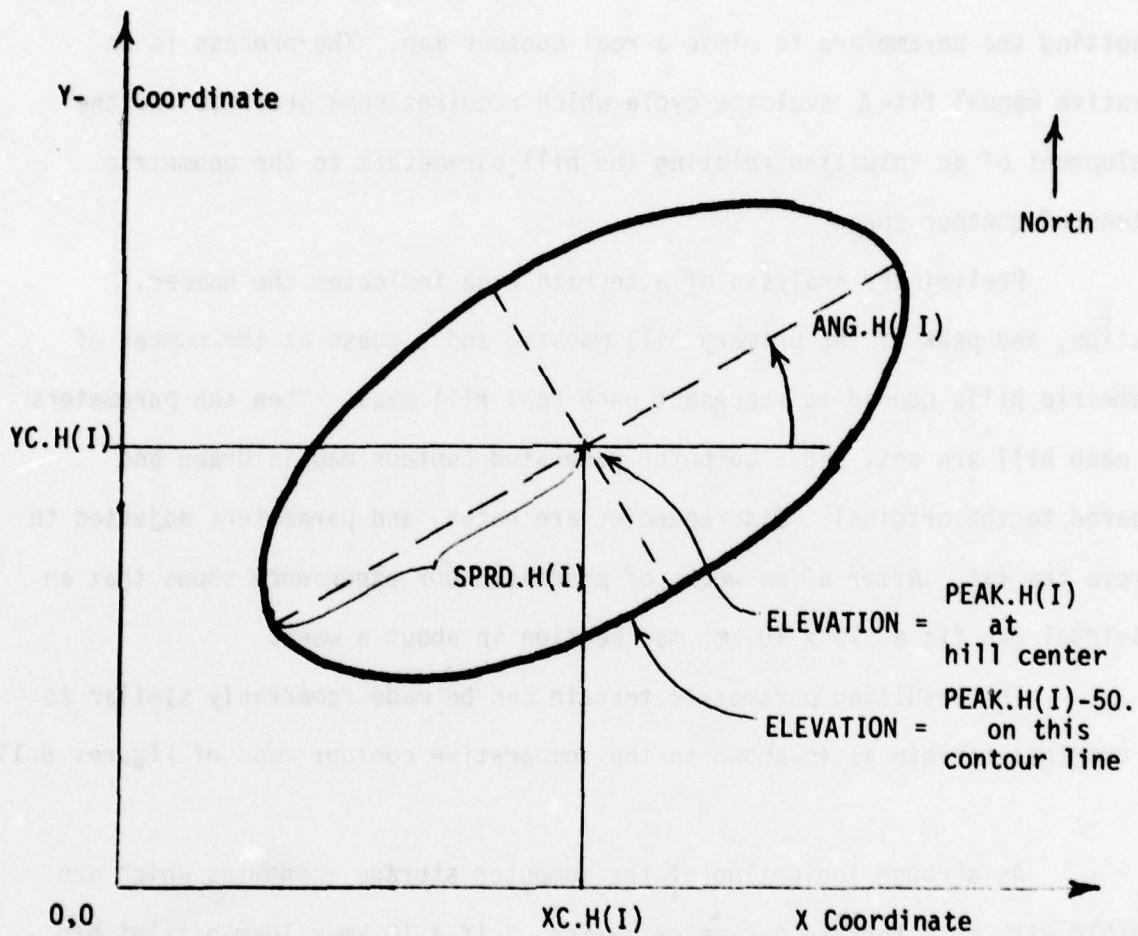


Figure 5. One elliptical contour of hill I

C. Fitting Hills to Real Terrain

Given the set of 8 hill parameters, (XC.H, YC.H, PEAK.H, ANG.H, ECC.H, SPRD.H, HT.H, CUT.H), our goal is to emulate a piece of real terrain by setting the parameters to mimic a real contour map. The process is an iterative manual fit- & -evaluate cycle which requires some practice and the development of an intuition relating the hill parameters to the geometric notions of contour shape.

Preliminary analysis of a terrain area indicates the number, location, and peak of the primary hill masses, and a guess at the number of parametric hills needed to represent each real hill mass. Then the parameters for each hill are set, and a computer generated contour map is drawn and compared to the original. Discrepancies are noted, and parameters adjusted to improve the fit. After a few weeks of practice, our experience shows that an individual can fit a 10 x 10 km map section in about a week.

The resulting parametric terrain can be made remarkably similar to the original terrain as is shown in the comparative contour maps of Figures 6-11.

As a rough indication of the computer storage economies which are possible with this terrain parameterization, if a 10 km x 10km battlefield can be represented using NHILLS = 100 hills, then 800 parameters must be stored. By contrast, a digitized terrain model working with a 100m grid size would have to store slightly more than 10,000 numbers. This represents a storage savings factor of about 12. A more careful comparison would have to consider the exact number of hills required as well as possibilities for storage packing in both models.

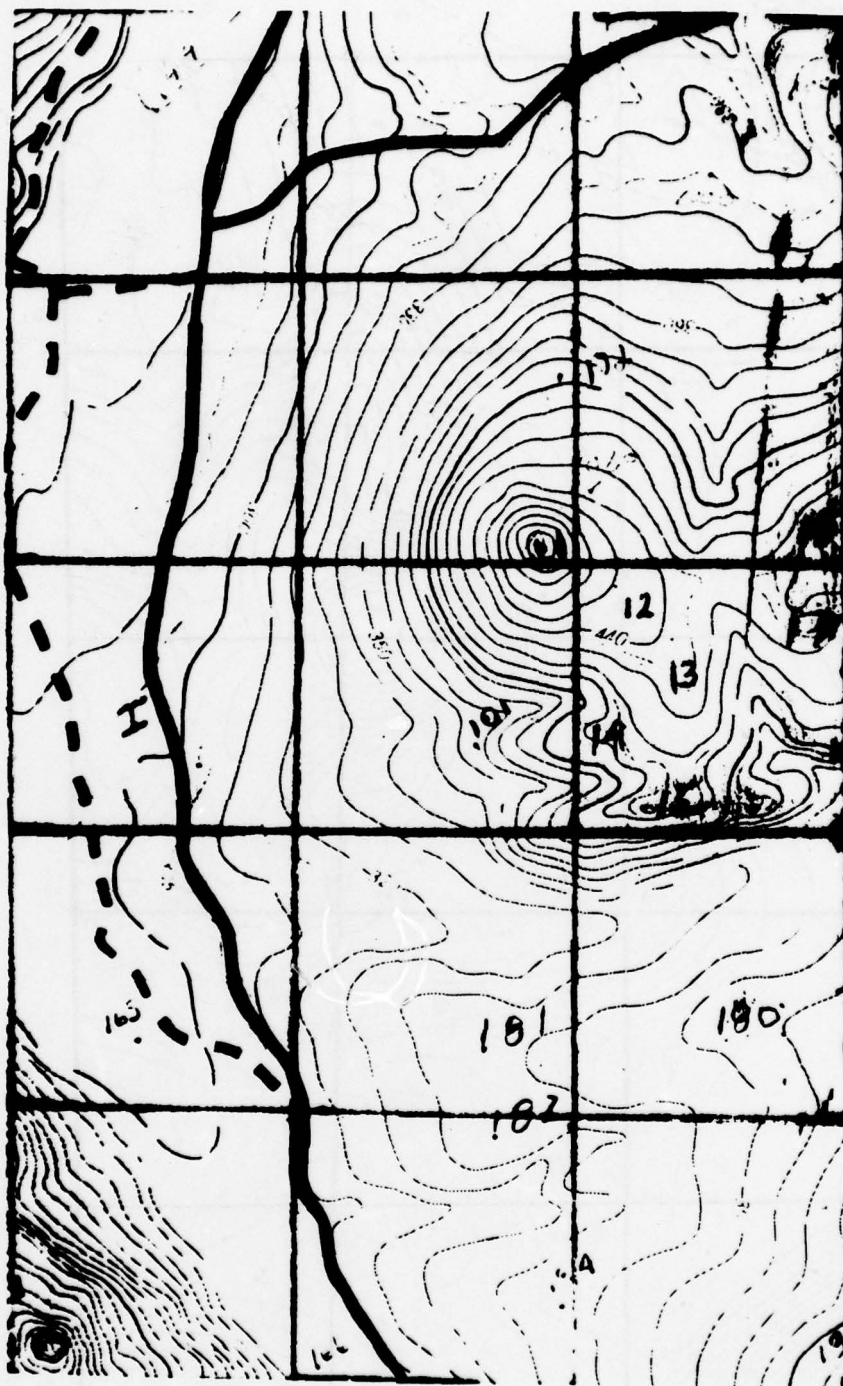


Figure 6. Map Contours A

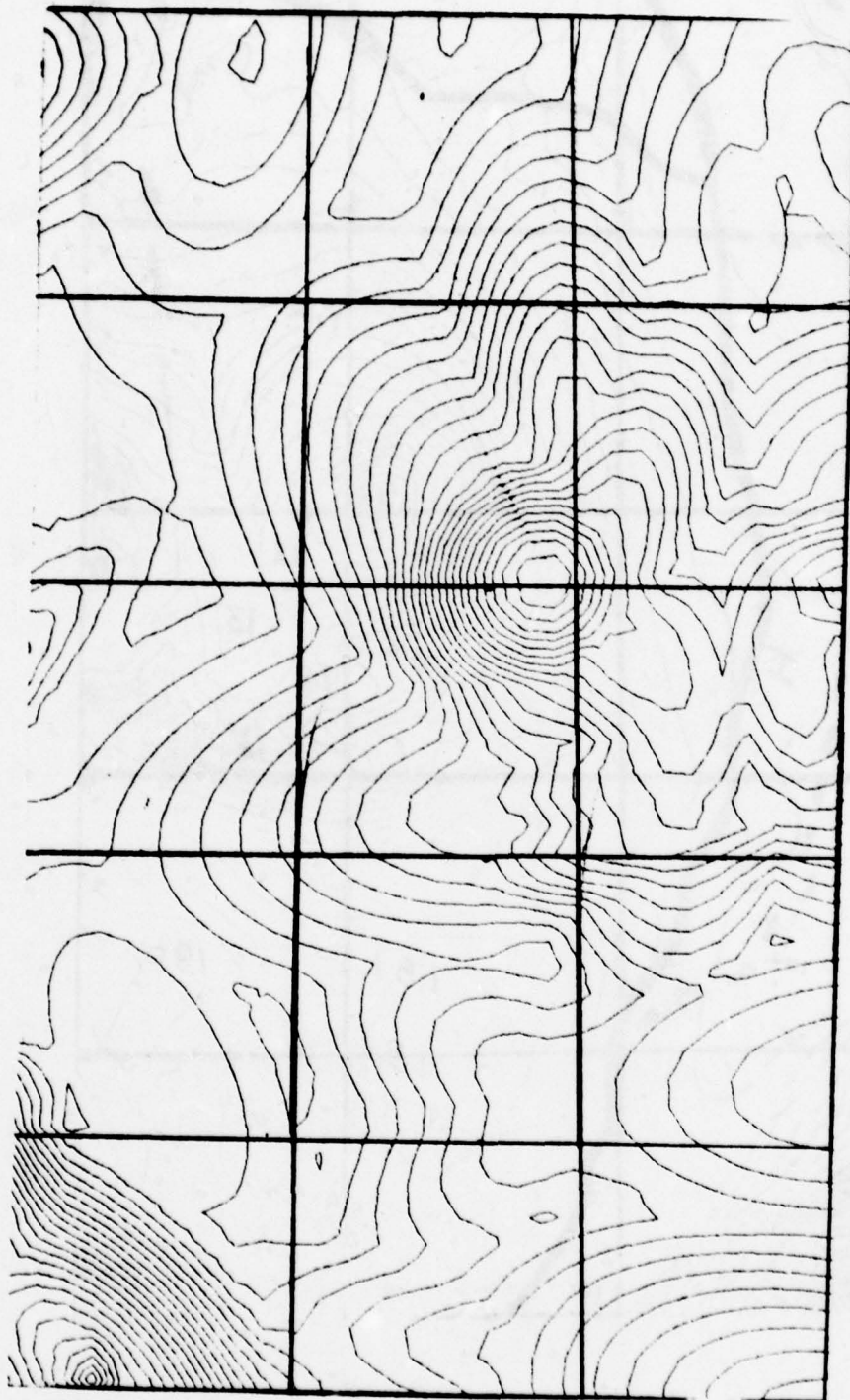


Figure 7. Parametric Contours A

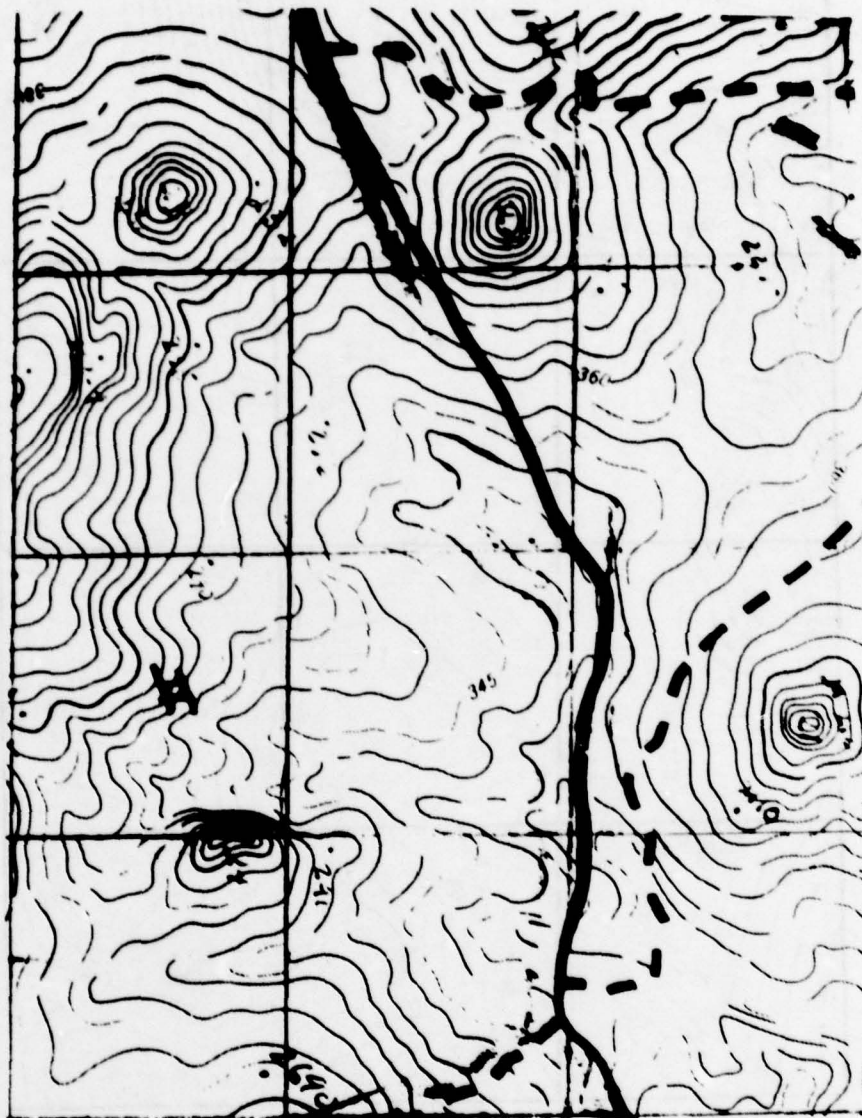


Figure 8. Map Contours B

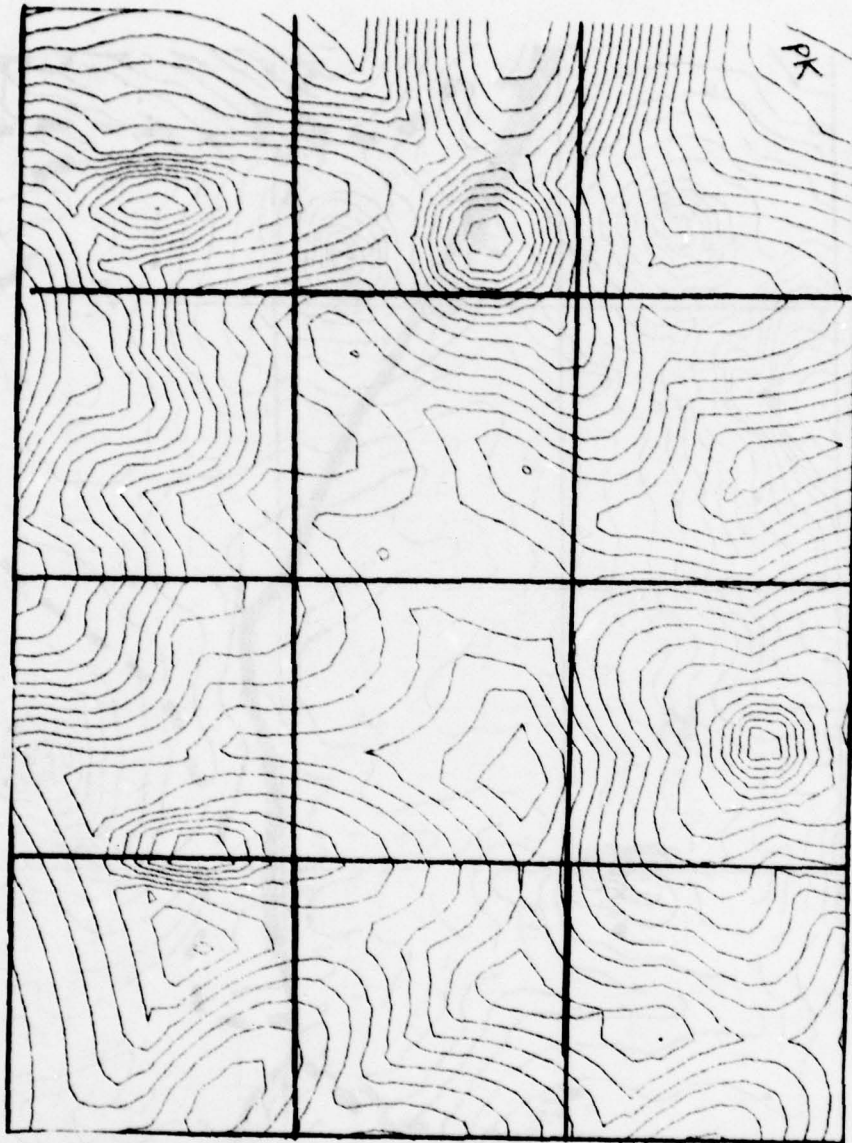


Figure 9. Parametric Contours B

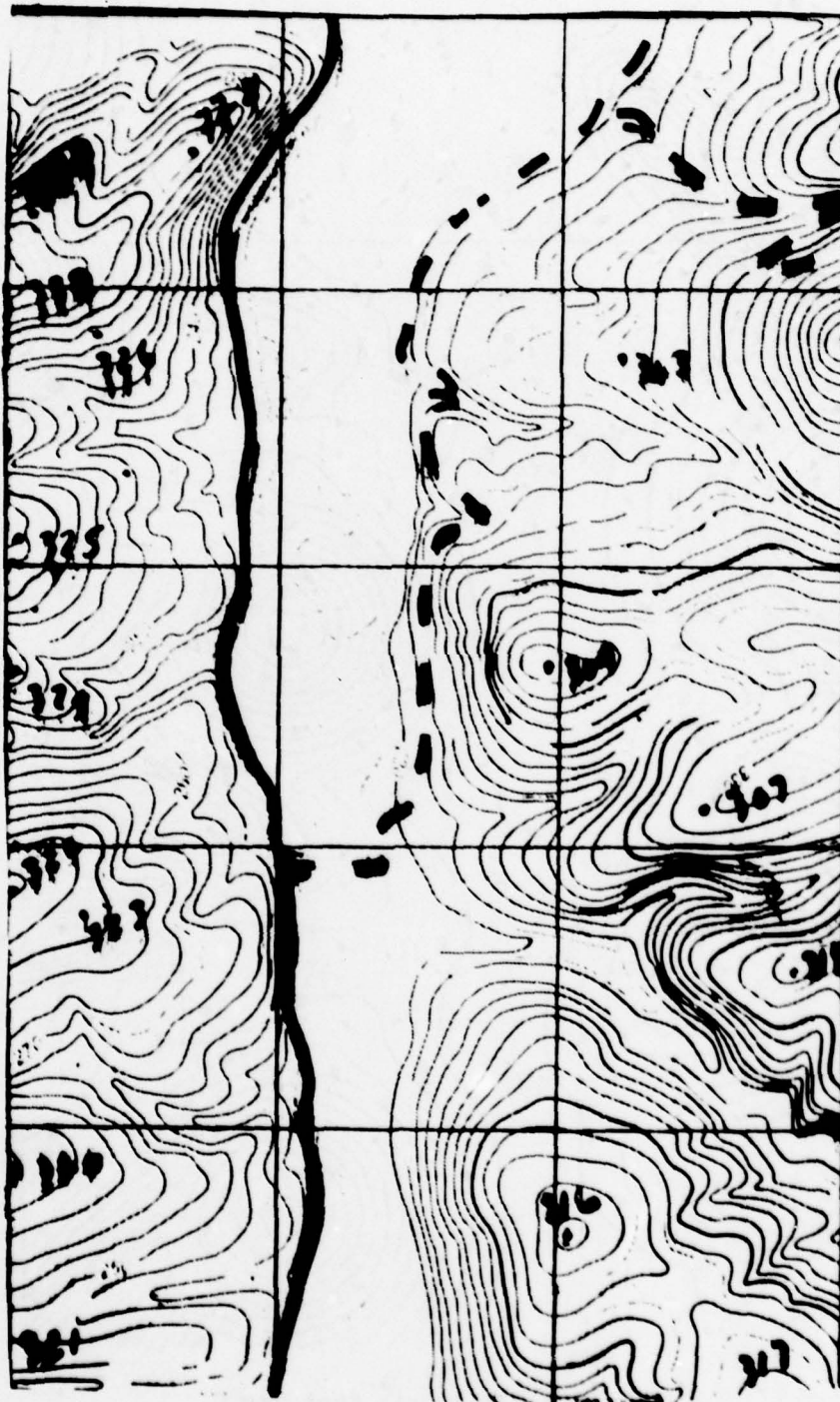


Figure 10. Map Contours C



Figure 11. Parametric Contours C

III. Elevation Computations

A. Parameterization for Computing

The fitting parameters discussed in Section II were designed for their geometric connotations to aid in the terrain fitting process. For computer calculations we are interested in computational efficiency rather than geometric appeal, so a second parameter set has been developed for the STAR terrain model. This set of computing-parameters is derived algebraically from the fitting parameters and yields exactly the same terrain elevations.

The basic elevation formula for hill I is, as in equation (2),

$$f_I(X,Y) = \text{PEAK.H(I)} - \text{HT.H(I)} + \text{HT.H(I)} * \exp [Q_I(X,Y)] \quad (5)$$

where $\text{PEAK.H(I)} - \text{HT.H(I)}$ gives the elevation of the bottom of the hill, and $\text{HT.H(I)} * \exp [Q_I(X,Y)]$ adds the hill mound to this bottom level as a function of X and Y.

For the computing parameterization, we define

$$\begin{aligned} XS &= X - XC.H(I) \\ YS &= Y - YC.H(I) \end{aligned} \quad (6)$$

giving the coordinate distances from X,Y to the hill center location. Then the quadratic function $Q_I(X,Y)$ is given as

$$Q_I(X,Y) = \text{PXX.H(I)} * XS^2 + \text{PYY.H(I)} * YS^2 + \text{PXY.H(I)} * XS * YS \quad (7)$$

where PXX.H(I) , PYY.H(I) , and PXY.H(I) are computing-parameters defined in terms of the fitting parameters as follows:

$$\begin{aligned} \text{Letting } SANG &= \sin (\text{ANG.H(I)}) \\ CANG &= \cos (\text{ANG.H(I)}) \\ A &= \ln (\text{HT.H(I)} / (\text{HT.H(I)} - 50.)) \\ B &= A * \text{ECC(I)}^2 \end{aligned} \quad (8)$$

$$\begin{aligned}
\text{then } PXX.H(I) &= - (A * CANG^2 + B * SANG^2) / SPRD.H(I)^2 \\
PYY.H(I) &= - (A * SANG^2 + B * CANG^2) / SPRD.H(I)^2 \\
PXY.H(I) &= (2 * CANG * SANG * (B - A)) / SPRD.H(I)^2
\end{aligned} \tag{9}$$

Finally, the CUT.H(I) parameter, which has been ignored in all of the above formulas is related to $Q_I(X,Y)$ by defining the computing parameter

$$CRIT.H(I) = \begin{cases} \ln ((HT.H(I) - CUT.H(I)) / HT.H(I)) & \text{if } CUT.H(I) < HT.H(I) \\ -\infty & \text{otherwise} \end{cases} \tag{10}$$

and redefining the basic hill equation to be

$$f_I(X,Y) = \begin{cases} PEAK.H(I) - HT.H(I) + HT.H(I) * \exp [Q_I(X,Y)] & \text{if } Q_I(X,Y) > CRIT.H(I) \\ BASE & \text{otherwise.} \end{cases} \tag{11}$$

The derivation of these formulas is rather tedious, but certainly not profound, consisting primarily of a coordinate rotation, the equation of an ellipse, and the definitions of SPRD.H and CUT.H. The reader who wishes to fully understand the mathematics of the terrain model should work through these derivations.

B. A Primitive ELEV Routine

Given the above computing parameters (XC.H, YC.H, PEAK.H, HT.H, PXX.H, PYY.H, PXY.H, CRIT.H) for each of a number NHILLS of hills on our battlefield, and given an overall terrain BASE, a primitive computer subroutine can easily be written to compute the elevation Z for any point X,Y on the battlefield. Such a program is given here in the SIMSCRIPT II.5 computer simulation language, assuming the above parameters have already been stored in appropriate global arrays and variables.

ROUTINE FOR PRIMITIVE.ELEV GIVEN X AND Y YIELDING Z

NORMALLY MODE IS REAL

DEFINE I AS AN INTEGER VARIABLE

LET Z = BASE

FOR I = 1 TO NHILLS DO

LET XS = X - XC.H(I) LET YS = Y - YC.H(I)

LET QI = PXX.H(I) *XS**2 + PYY.H(I) *YS**2 + PXY.H(I)*XS*YS

IF QI LT CRIT.H(I) CYCLE

ELSE LET FI = PEAK.H(I) + HT.H(I)*(EXP.F(QI)-1.)

IF FI GT Z LET Z = FI ALWAYS

LOOP RETURN END

Given the formulas in Section III.A, this routine should not require further documentation. It is the prototype for the ELEV routine actually used in STAR.

C. Refinements for Speeding the Calculations

The PRIMITIVE.ELEV routine, while it will work for any X,Y, is not particularly efficient. Its primary weakness is the need to loop over all NHILLS hills even though only a few hills are close enough to a given X,Y to have any chance of influencing the macro-terrain elevation Z at that point. Substantial economies in computing time can be realized by using a terrain preprocessor program to develop lists of the hills which are relevant to particular areas of the battlefield and to improve CUT.H values for all hills. The terrain preprocessor program, called HILL.LIST, divides the battlefield into grid squares of size GSIZE (our experience indicates that GSIZE in the range 1 to 3 km yields an effective grid). Each grid square is referenced by grid subscripts IX and IY. An array LIST.H (IX,IY,L) is created which contains, for each IX,IY (i.e. for each grid square) a list of the hills I which

contribute to the terrain in the grid square.

Then given X, Y we can compute the elevation Z by first computing the appropriate grid square IX, IY and then executing a routine similar to `PRIMITIVE.ELEV` which only loops over the hills in `LIST.H(IX, IY, L)` for this grid square. Dramatic computational efficiencies result with only a modest storage increase for the list. Typical results from our experience are that for $GSIZE = 1$ km., on the average, 4 or 5 hills are relevant to each grid square. Clearly this depends on the complexity of the terrain and the number of hills used to model it.

A further savings can be obtained by computing in `HILL.LIST` the elevation E of the lowest terrain point actually represented by each hill and using E with `PEAK.H` to get the smallest `CUT.H` value possible for the hill. Then the `QI vs CRIT.H(I)` test in `ELEV` will cycle more frequently hence avoiding the expensive exponential computation.

The line-by-line details of `HILL.LIST` are documented in Section VII. For this section it suffices to define the global `LIST.H (IX, IY, L)` array which `HILL.LIST` creates. `LIST.H` has subscripts

IX : grid square subscript in X direction

$IX = 1, \dots, NGRIDX$

IY : grid square subscript in Y direction

$IY = 1, \dots, NGRIDY$

L : subscript for last coordinate of `LIST.H`.

For a given IX, IY ,

`LIST.H(IX, IY, 1)` = BASE value for the grid square

`LIST.H(IX, IY, 2)`, `LIST.H(IX, IY, 3)`, ETC. = hill numbers for the hills which influence terrain in this gridsquare.

LIST.H is represented in SIMSCRIPT as a ragged array, so the number of hills to be scanned in grid square IX,IY is DIM.F(LIST.H(IX,IY,*))-1 .

The resulting streamlined ELEV routine is presented in Listing 1.

Local variables

I hill number

IX } grid square index in X and Y directions
IY }

KOUNT number of hills in a grid square + 1

L loop index

X } input X and Y coordinates
Y }

Z resulting elevation

XS,YS,QI,FI intermediate computations as in PRIMITIVE.ELEV

Global variables

X.LO.BDRY } map coordinates of southwest corner of battlefield in meters
Y.LO.BDRY }

GSIZE grid square size in meters

NGRIDX range of IX - number of grid squares in X direction

NGRIDY range of IY - number of grid squares in Y direction

LIST.H list of hill numbers for each grid, also BASE value

DUM.I one-dimensional integer dummy array to simplify
coordinate computation in accessing LIST.H

XC.H,YC.H,PEAK.H,HT.H,PXX.H,PYY.H,PXY.H,CRIT.H hill parameter arrays.

Routines called

None

Events scheduled

None

```

1 ROUTINE FOR ELEV GIVE X,Y YIELDING Z
2 **ROUTINE TO COMPUTE ELEVATION Z FOR GIVEN X,Y COORDINATES
3 DEFINE I,X,Y,KOUNT,L AS INTEGER VARIABLES
4 DEFINE X,Y,Z,AS,YS,UI,FI AS REAL VARIABLES
5 LET IX=I+TRUNC.F((X-X.LC.BORY)/GSIZE)
6 LET IY=I+TRUNC.F((Y-Y.LC.BORY)/GSIZE)
7 IF IX LT 1 LET IX=1 ALWAYS
8 IF IX GT UGRIDX LET IX = UGRIDX ALWAYS
9 IF IY LT 1 LET IY=1 ALWAYS
10 IF IY GT UGRIDY LET IY = UGRIDY ALWAYS
11 LET DUM.I(4) = LIST.H(IX,IY,4) **DUMMY ARRAY TO SIMPLIFY INDEXING
12 LET KOUNT = DIM.F(DUM.I(4))
13 LET Z = DUM.I(1)
14 FOR L = 2 TO KOUNT DO
15   LET I = DUM.I(L)
16   LET AS=X-XC.F(I)
17   LET UI=PX.A.H(I)*AS*AS + PYY.H(I)*YS*YS + DXY.H(I)*AS*YS
18   IF UI LT CRIT.H(I) CYCLE
19   ELSE LET FI=PEAK.H(I)*FT.H(I)*(EXP.F(0.1)-1.)
20   IF FI GT Z LET Z=FI ALWAYS
21   LOOP
  RETURN  INI

```

ELEV Listing 1

THIS PAGE BEST QUALITY PRACTICABLE
FROM GO... FISHED TO DDC

Lines 5-10 compute the gridsquare indices and make sure that the gridsquare is on the battlefield.

Lines 11-12 define DUM.I to be the appropriate column of LIST.H and determine the length of that column

Line 13 sets Z to the BASE value

Lines 14-20 loop over the hills I (for this grid square only) and increase Z if the computed hill elevation FI is greater than the max so far.

Note that the origin for the grid square reference system is at X.LO.BDRY, Y.LO.BDRY (the southwest corner of the battlefield).

D. Terrain Slope

In addition to terrain elevation, the slope of the terrain at any point X,Y is directly available. Differentiating equation (11) gives the gradient components

$$GX = \frac{\partial f_I(X,Y)}{\partial X} = \begin{cases} HT.H(I) \cdot \exp[Q_I(X,Y)] * \frac{\partial Q_I(X,Y)}{\partial X} & \text{if } Q_I(X,Y) > CRIT.H(I) \\ 0 & \text{otherwise} \end{cases} \quad (12)$$

where

$$\frac{\partial Q_I(X,Y)}{\partial X} = 2 * PXX.H(I) * XS + PXY.H(I) * YS \quad (13)$$

and

$$GY = \frac{\partial f_I(X,Y)}{\partial Y} = \begin{cases} HT.H(I) \cdot \exp[Q_I(X,Y)] * \frac{\partial Q_I(X,Y)}{\partial Y} & \text{if } Q_I(X,Y) > CRIT.H(I) \\ 0 & \text{otherwise} \end{cases} \quad (14)$$

where

$$\frac{\partial Q_I(X,Y)}{\partial Y} = 2 * PYY.H(I) * YS + PXY.H(I) * XS \quad (15)$$

Then if we move away from X,Y in direction ΔX , ΔY the slope is given by the directional derivative

$$D = (GX * \Delta X + GY * \Delta Y) / \sqrt{\Delta X^2 + \Delta Y^2} . \quad (16)$$

An elevation routine called ELEVG which computes the gradient components as well as Z is available and is used by the STAR movement routines. A separate routine is used because the line of sight routine which calls ELEV does not need the gradient information and should not pay the added computational price. The computational sequence in ELEVG is slightly different to capitalize on common computations in Z, GX, and GY. The resulting code is given in Listing 2. Because of the similarity to ELEV we will not provide a line-by-line explanation of ELEVG.

IV. Forest Modelling

A. Brief Description of Forest Methodology

In addition to the macro-terrain, another factor which crucially influences line of sight computations in the STAR model (as in the real world) is the presence of forested areas. Although we will refer to "forests" throughout this discussion, the modeling tools developed here could just as well be applied to man-made features, such as towns, which provide cover and thus interrupt LOS.

The same basic ideas can also be expanded to model smoke clouds on the battlefield. The resulting clouds must be able to move, expand, and disperse (which forests do not do), and are not totally opaque to all sensor systems. Development of a smoke module for STAR is currently underway, but it will not be documented here.

Forests or other cover features in the STAR model are represented by elliptical areas on the ground. Each such cover ellipse has a tree height

```

1 ROUTINE FOR EVEVG GIVEN X,Y YIELDING Z,GX,GY
2 **ROUTINE TO COMPUTE ELEVATION Z AND GRADIENT
3 DEFINE I,XA,IV,KOUNT,1 AS INTEGER VARIABLES
4 DEFINE X,Y,Z,GX,GY,XS,YB,LA,XY,QXY,OI,TERM,FI AS REAL VARIABLES
5 LET IX=1+TRUNC(FIX(A-A.LO,HPRV)/GSIZE)
6 LET IV=1+TRUNC(FIX(Y-Y.LO,HPRV)/GSIZE)
7 IF IX LT 1 LET IX=1 ALWAYS
8 IF IX GT NGRIDX LET IX = NGRIDX ALWAYS
9 IF IV LT 1 LET IV=1 ALWAYS
10 IF IV GT NGRIDY LET IV = NGRIDY ALWAYS
11 LET DUM,I(*) = LIST.F(IX,IV,*) DUMMY ARRAY TO SIMPLIFY INDEXING
12 LET KOUNT = DIM.F(DUM,I(*))
13 LET Z = DUM.I(1)
14 FOR L = 2 TO KOUNT DO
15   LET XS=A-XC.H(I)
16   LET GX = FXXA.H(I)*XS
17   LET OXY = PXXA.H(I)*XS
18   LET OI = OAXS + VXS*(GY+QXY)
19   IF OI LT CRIT.H(I) CYCLE
20   ELSE LET TERM = HI.H(I)*EXP.F(OI)
21   LET FI = FFAK.F(I)+TERM-HI.H(I)
22   IF FI GT Z LET Z = FI
23   LET GX = TERM*(Z-XC + OXY.H(I)*YS)
24   LET GY = TERM*(Z-YC + OXY)
25   LET YS=Y-YC.H(I)
26   LET OY = FYY.F(I)*YS
27 LOOP RETURN END

```

EVEVG
Listing 2

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDO

associated with it, and the forest is thus an elliptical "cylinder" with that fixed height above the macro terrain. Forests with non-elliptical shapes and non-constant heights can be approximated by combining several possibly overlapping ellipses. The tree height at a given point X,Y is the maximum tree height for all the forest ellipses containing the point X,Y .

B. Parameterization for Forest Fitting

As is the case for macro terrain hill fitting, a geometrically motivated parameterization of the cover ellipses has been developed to aid in the fitting of the model to terrain maps. The parameters are listed below and illustrated in Figure 12.

- XC.E(I): the X coordinate of the ellipse center
- YC.E(I): the Y coordinate of the ellipse center
- ANG.E(I): the orientation angle measured counterclockwise
in degrees from east to the major axis of the ellipse
- AMAJ.E(I): length in meters of the semi-major axis of the ellipse
- AMIN.E(I): length of semi-minor axis of the ellipse
- HT.E(I): height of the trees in ellipse I above the macro terrain elevation.

C. Parameterization for Computing

Given an arbitrary point X,Y on the battlefield we want to determine if this point is inside a forest feature, and, if so, how high the trees are. Since this computation will be performed frequently in the LOS procedure, it is important to have an efficient computational procedure. Thus the ellipses, like the hills, have a transformed parameter set for computing.

The boundary of the I^{th} forest ellipse is represented by the quadratic equation

```

1  SUBROUTINE FIVE (X, Y, Z, GX, GY, GZ, XG, YG, ZG, XG, YG, ZG, XG, YG, ZG)
2  * ROUTINE TO COMPUTE THE COORDINATES OF THE CENTER OF GRAVITY
3  * OF A BODY FROM THE COORDINATES OF THE CORNERS
4  * OF THE BODY
5  *
6  * INPUT: X, Y, Z, GX, GY, GZ, XG, YG, ZG, XG, YG, ZG, XG, YG, ZG
7  * OUTPUT: X, Y, Z, GX, GY, GZ, XG, YG, ZG, XG, YG, ZG, XG, YG, ZG
8  *
9  *
10 *
11 *
12 *
13 *
14 *
15 *
16 *
17 *
18 *
19 *
20 *
21 *
22 *
23 *
24 *
25 *
26 *
27 *

```

EVEVG
Listing 2

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO HQ

associated with it, and the forest is thus an elliptical "cylinder" with that fixed height above the macro terrain. Forests with non-elliptical shapes and non-constant heights can be approximated by combining several possibly overlapping ellipses. The tree height at a given point X,Y is the maximum tree height for all the forest ellipses containing the point X,Y .

B. Parameterization for Forest Fitting

As is the case for macro terrain hill fitting, a geometrically motivated parameterization of the cover ellipses has been developed to aid in the fitting of the model to terrain maps. The parameters are listed below and illustrated in Figure 12.

- XC.E(I): the X coordinate of the ellipse center
- YC.E(I): the Y coordinate of the ellipse center
- ANG.E(I): the orientation angle measured counterclockwise
in degrees from east to the major axis of the ellipse
- AMAJ.E(I): length in meters of the semi-major axis of the ellipse
- AMIN.E(I): length of semi-minor axis of the ellipse
- HT.E(I): height of the trees in ellipse I above the macro terrain elevation.

C. Parameterization for Computing

Given an arbitrary point X,Y on the battlefield we want to determine if this point is inside a forest feature, and, if so, how high the trees are. Since this computation will be performed frequently in the LOS procedure, it is important to have an efficient computational procedure. Thus the ellipses, like the hills, have a transformed parameter set for computing.

The boundary of the I^{th} forest ellipse is represented by the quadratic equation

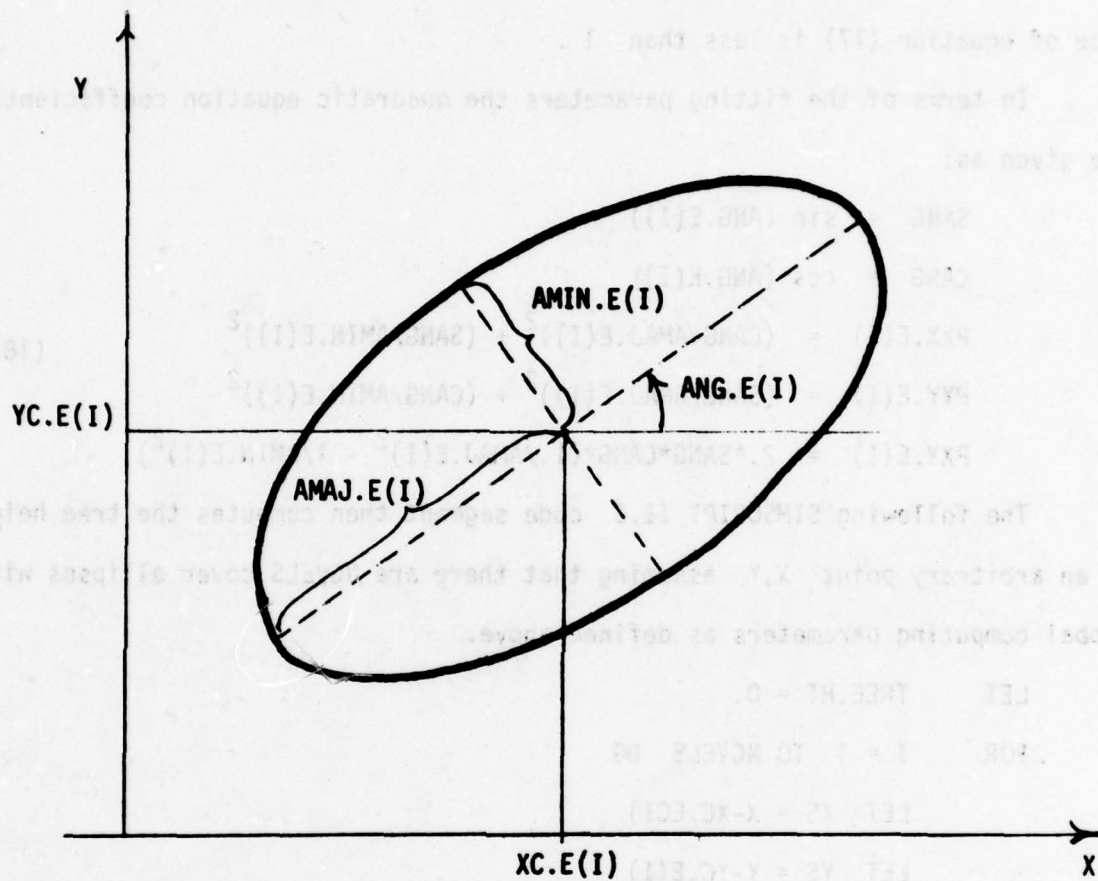


Figure 12. Forest Ellipse Fitting Parameters

$$\begin{aligned} &PXX.E(I)*(X-XC.E(I))^2 + PYY.E(I)*(Y-YC.E(I))^2 \\ &+ PXY.E(I)*(X-XC.E(I))*(Y-YC.E(I)) = 1.0 \end{aligned} \quad (17)$$

Then an arbitrary X,Y is inside the ellipse if and only if the left hand side of equation (17) is less than 1 .

In terms of the fitting parameters the quadratic equation coefficients are given as:

$$\begin{aligned} SANG &= \sin (ANG.E(I)) \\ CANG &= \cos (ANG.E(I)) \\ PXX.E(I) &= (CANG/AMAJ.E(I))^2 + (SANG/AMIN.E(I))^2 \\ PYY.E(I) &= (SANG/AMAJ.E(I))^2 + (CANG/AMIN.E(I))^2 \\ PXY.E(I) &= 2.*SANG*CANG*(1./AMAJ.E(I)^2 - 1/AMIN.E(I)^2) \end{aligned} \quad (18)$$

The following SIMSCRIPT II.5 code segment then computes the tree height at an arbitrary point X,Y assuming that there are NCVELS cover ellipses with global computing parameters as defined above.

```
LET    TREE.HT = 0.
FOR    I = 1 TO NCVELS DO
    LET XS = X-XC.E(I)
    LET YS = Y-YC.E(I)
    LET QI = PXX.E(I)*XS**2 + PYY.E(I)*YS**2
           + PXY.E(I)*XS*YS
    IF   QI GE 1.  CYCLE
    ELSE IF HT.E(I) GT TREE.HT LET TREE.HT = HT.E(I)
    ALWAYS
```

LOOP

D. Efficiency

As is the case for terrain hills, greater efficiency can be obtained

by limiting the number of cover ellipses checked at each X,Y to those found in a grid square. A forest preprocessor program called TREE.LIST operates in a fashion exactly analogous to HILL.LIST to develop an array LIST.C(IX,IY,L) of cover ellipse numbers in grid square IX,IY. Again this is stored as a SIMSCRIPT ragged array. The resulting more efficient code segment differs from the above code just as in the ELEV program. Since it appears in the LOS program we will not detail it further here beyond defining the LIST.C array contents. For given IX,IY

LIST.C(IX,IY,1) = number of forest ellipses overlapping
grid square IX,IY.

and

LIST.C(IX,IY,2),...etc. give the ellipse numbers if
LIST.C(IX,IY,1) ≥ 1 .

Documentation of the TREE.LIST program appears in Section VII.

V. Overview of LOS Modelling in STAR

An important computation for any high resolution combat simulation is the line of sight (LOS) routine. The basic problem is the following: Given an observer (A) and a potential target (B), what part (if any) of the target can be seen by the observer? LOS is a purely geometric computation in that we assume perfect visibility. Degraded visibility conditions are incorporated in other modules of the STAR model. The result of the LOS computation is a percent of the vertical height of the target B visible to the observer A. This percent visible is used in various ways in other parts of the STAR model.

The LOS computations are rather complex, but conceptually they are based on a simple procedure: "Find the lowest sight line from A over the terrain (and forests). Extend this line to B's location, and compare its extrapolated height to B's elevation. Thus compute % visible." This

procedure is illustrated in figure 13 for two cases. In the first, a terrain hill blocks LOS completely so 0% of the target is visible. In the second case the lowest sight line intersects the target so that 40% of its vertical height can be seen by A while the other 60% is covered.

This simple concept is rather difficult to compute because

1. any one of the hills or forests which lie between A and B may be the feature which determines the lowest sight line. Thus we have to repeat computations for each such feature, and
2. the equation for the lowest sight line over a hill does not have a closed form solution.

The following observations tend to lessen the above computational difficulties:

3. Since A and B are given locations, only hills and forests which intersect the line joining A and B need to be considered. Features which are remote from this line can be ignored totally.
4. If LOS does not exist at all, (0% visible) then as soon as a hill or forest has been found which proves that the target is 0% visible, the computation can stop. All other hills and forests can be ignored.

Point 4. is particularly significant because on a typical battlefield a large fraction of the potential observer-target pairs will not have line of sight. Since the lowest sight line computation is difficult, the STAR LOS routine first performs a number of simpler tests trying to prove either that % visible = 0, or that a particular hill is so small that it could not possibly interrupt LOS between A and B. In either case the difficult lowest sight line computation is avoided, and in the first case all further

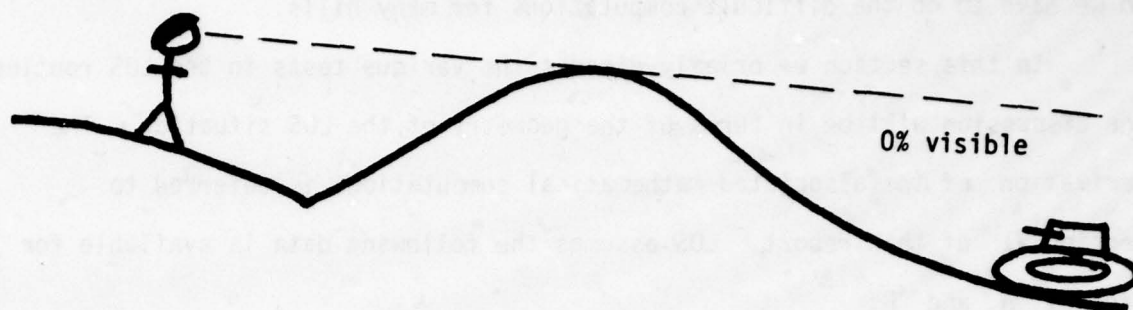


Figure 13. Lowest Sight Line Determines % Visible

computations are unnecessary. Only in cases where % visible is non zero do we have to do the difficult computations for many hills.

In this section we briefly discuss the various tests in the LOS routine. The discussion will be in terms of the geometry of the LOS situation. The derivation of the associated mathematical computations is deferred to Section VI of this report. LOS assumes the following data is available for each of A and B:

XA,YA(XB,YB) The X and Y coordinates on the
battlefield for each of A and B .

TMACA,(TMACB) The macro terrain elevation computed
from the ELEV routine.

TMICA (TMICB) A micro terrain offset + or - from
the macro terrain. This offset is used in
several ways:

- i) For stationary defenders in defilade,
negative micro terrain offsets simulate
the defilade position.
- ii) For elements in the open, a random micro
terrain choice can be used to simulate
terrain irregularities that are too small
to explicitly model in the ELEV routine.
- iii) Large positive micro terrain elevations
are used to simulate aircraft flying
above the terrain.

SIZEA (SIZEB) The vertical height dimension of each element.
The percent visible is taken as a fraction of
this size.

LAGA (LAGB) Indicator variables for air/ground

0 \Rightarrow the element is on the ground

1 \Rightarrow the element is an aircraft

LATOB (LBTOA) Indicator variables for one or two way LOS calls.

LATOB = 1 \Rightarrow compute LOS from A to B

(yielding VISFRB)

0 \Rightarrow do not compute A to B

Since many of the situations in which LOS is computed are two-way situations, (Eg. A is trying to detect B, but B is also trying to detect A) the routine includes the option to do all calculations in both directions, thus saving repetition of many common computations. The result of the LOS computations is provided in the two variables

VISFRA (VISFRB) The fraction of SIZEA (SIZEB) which can be seen by B(A)

To compute LOS we evaluate terrain and forest height at a number of points along the line segment between A and B. The various tests are sequenced in the same order in the computer code as in the discussion which follows. For this discussion we will consider a one-way LOS CALL with A as observer and B as target (LATOB=1, LBTOA = 0). The two way computation is exactly analogous.

A. Initialize

To begin with, the LOS routine sets the visible fraction VISFRB=1.0 . As we proceed through the routine, various tests may decrease the visible fraction, but none will ever increase it. If VISFRB ever decreases to 0, then we immediately terminate the computations and return from the routine.

Throughout the computations we assume that the bottom of A is at $TMACA + TMICA$ and that the top of A is at $ZA = TMACA + TMICA + SIZEA$ and similarly for B (see Figure 14). The observation device is assumed to be at the top of A. If the micro terrain value $TMICB$ is negative, then it is assumed that the bottom part of the target is not visible, so in this case $VISFRB$ is immediately decreased to

$$VISFRB = 1. + \frac{TMICB}{SIZEB} \quad (19)$$

which is less than 1 when $TMICB < 0$.

B. Grid Square List

The line segment between A and B's positions crosses over some of the terrain grid squares. Thus the only hills and forests we need to consider are those associated with these grid squares. The routine develops a list of the NGRSQ grid squares crossed in the arrays $IGX(K)$ and $IGY(K)$ for $K=1, \dots, NGRSQ$.

C. Forest Ellipse List

The line segment between A and B may intersect some of the forest cover ellipses. If so, we must know where the intersection points lie. The routine develops a list of these ellipse intersections by considering, for each grid square only the ellipses (in $LIST.C$) which appear in the grid, and then testing each for intersection with the A to B line. The intersection points (if they exist) are called $S1$ and $S2$ and are stored along with the ellipse numbers in arrays $IEL(K)$, $CS1(K)$ and $CS2(K)$ for $K=1, \dots, NELS$. Although an ellipse may appear in several adjacent grid squares, it will be stored on the list at most one time.

D. LOS Test at Forest Boundaries

One point at which lack of LOS can often be proved is a forest

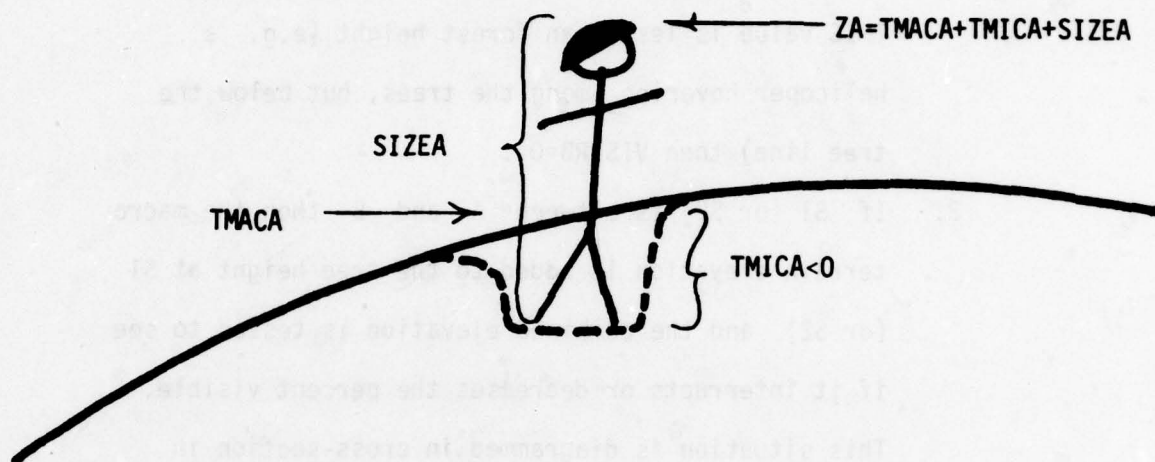


Figure 14. Observer Geometry

boundary, so as the above list of forest ellipses is being developed LOS is tested in the following ways:

1. If A or B is a ground element and is inside a forest ellipse, VISFRB=0. (no visibility through forests)
2. If A or B is an air element but the corresponding TMIC value is less than forest height (e.g. a helicopter hovering among the trees, but below the tree line) then VISFRB=0 .
3. If S1 (or S2) is between A and B then the macro terrain elevation is added to the tree height at S1 (or S2) and the combined elevation is tested to see if it interrupts or decreases the percent visible. This situation is diagrammed in cross-section in Figure 15. Subroutine TREE.CHECK performs these computations.

E. HILLTOPS

Next we consider all macro terrain hills that might lie between A and B. The list of such hills is developed by scanning LIST.H for each grid square crossed by the A to B line. Each such hill is processed one at a time from this point through to the end of the computations. Although a hill may appear in several grid squares, each hill is considered only once. We first compute a point W on the A to B line which is the top of the hill cross-section cut by that line. The following tests are then done on W.

1. If W is far from being between A and B, then this hill is irrelevant; go on to the next hill (see Figure 16, part 1).

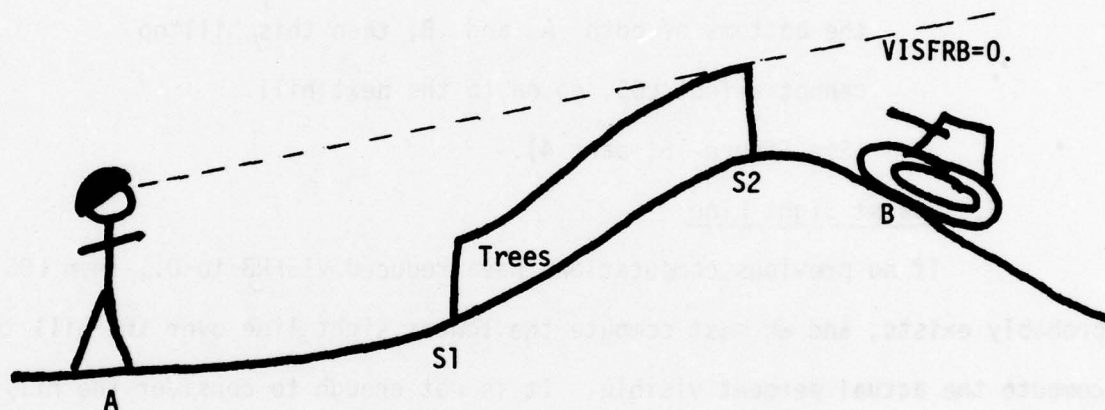


Figure 15. LOS Test at Edge of Forest

2. If the hill height at W is negligible (i.e. the cross section is far from the hill center) then this hill is irrelevant; go on to the next hill. (See Figure 16, part 2).
3. Test the hill height at W plus the tree height at W (if any) to see if LOS is totally interrupted. (See Figure 16, part 3).
4. If hill height at W plus the tree height is below the bottoms of both A and B, then this hilltop cannot affect LOS, go on to the next hill. (See Figure 16, part 4).

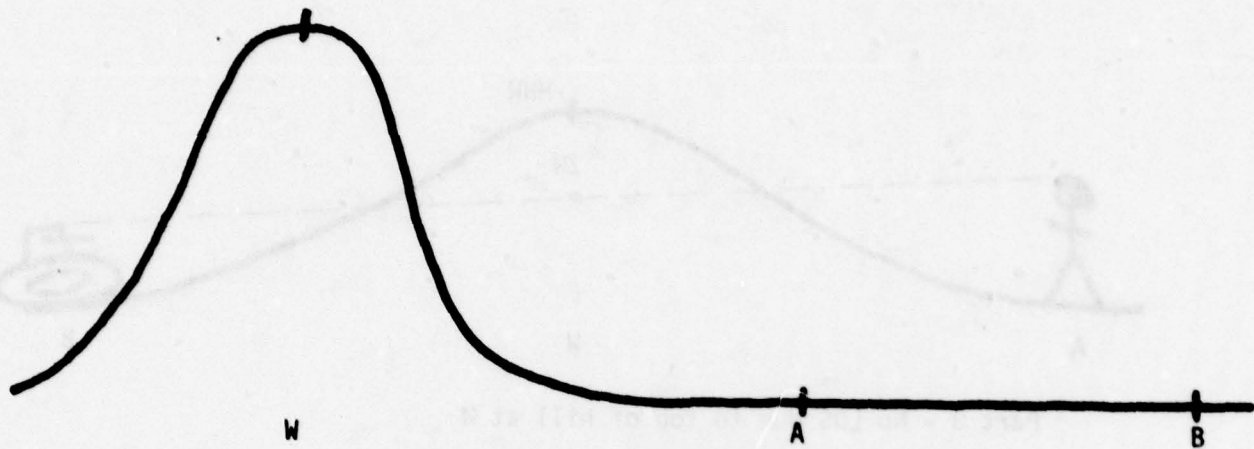
F. Lowest Sight Line

If no previous computations have reduced VISFRB to 0., then LOS probably exists, and we must compute the lowest sight line over the hill to compute the actual percent visible. It is not enough to consider the hilltop, since for cases where A is significantly above or below the hilltop, the lowest sight line will graze the hilltop away from W. (See Figure 17). We denote by V the point at which the lowest sight line is tangent to the hill. Once V is found, then the lowest sight line is extrapolated to B's position for the computation of VISFRB (as in Figure 13). Any forest coverage at V is, of course, also considered.

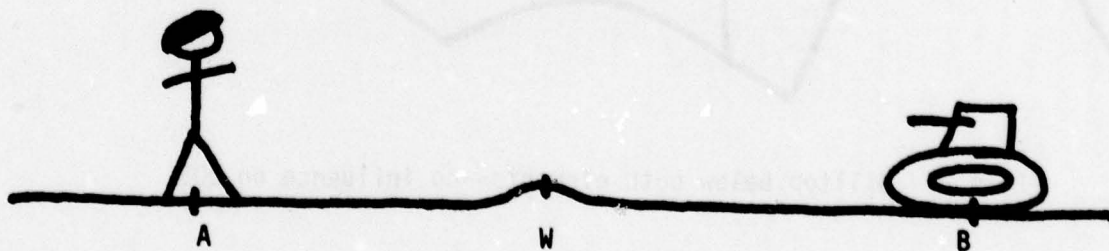
When these computations have been done for each hill, with VISFRB possibly decreased at each step, the final resulting VISFRB value is returned as the result of the LOS routine.

VI. Details of the Line of Sight Computations

In this section we derive the mathematical basis for the computations

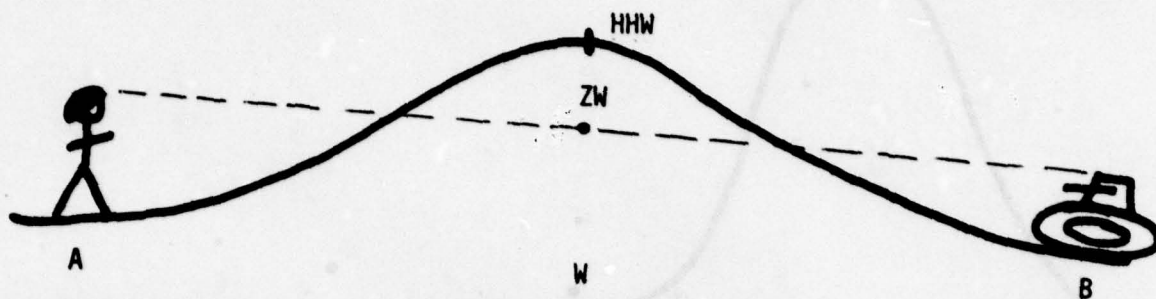


Part 1 - Hilltop is remote--no influence on LOS

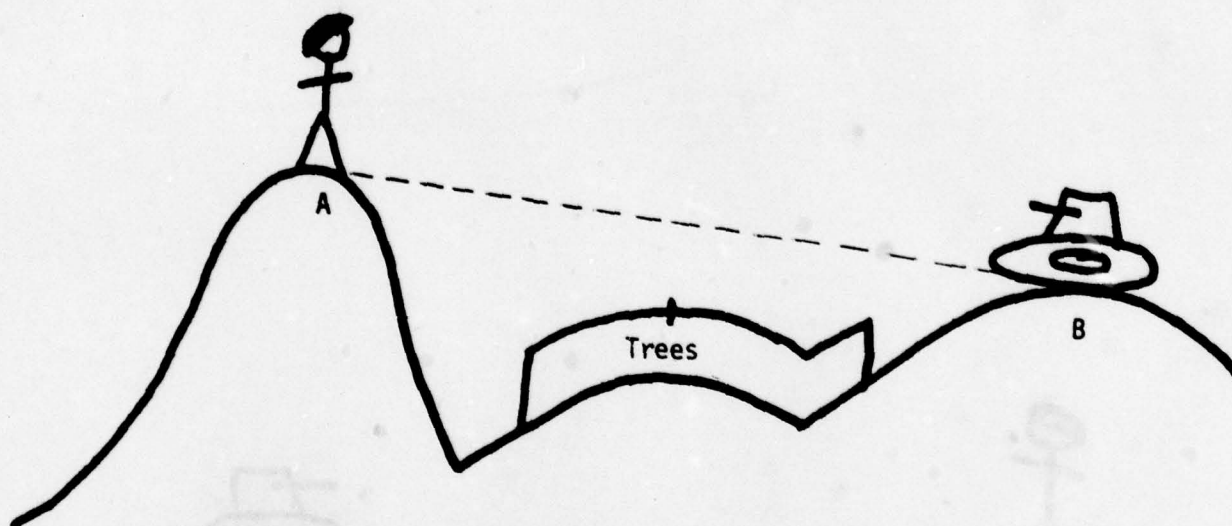


Part 2 - Hilltop is of negligible height--no influence on LOS

Figure 16. Cases to consider at W.
(1 & 2)



Part 3 - No LOS due to top of hill at W



Part 4 - Hilltop below both elements--no influence on LOS

Figure 16 (3 & 4). (continued) Cases at W

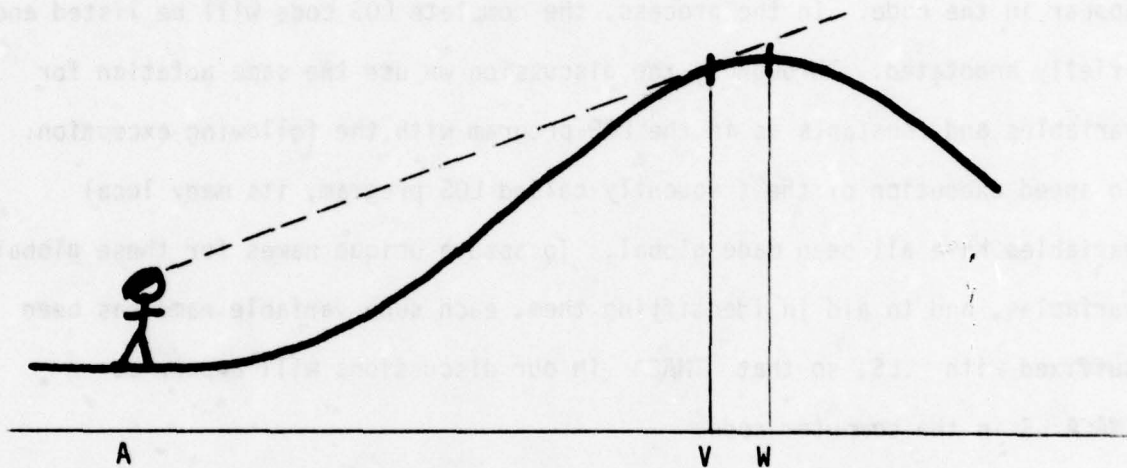


Figure 17. Lowest Sight Line Not at W

in the LOS routine and examine the SIMSCRIPT LOS code line by line. This section is the most technical of the report, and the following sections can be read without comprehending all of this material. Thorough understanding of this section is, however, required for anyone who wants to adjust or modify the STAR LOS procedure. The computations will be detailed in the same order as they were discussed in Section V, and in essentially the same order as they appear in the code. In the process, the complete LOS code will be listed and briefly annotated. Throughout the discussion we use the same notation for variables and constants as in the LOS program with the following exception: To speed execution of the frequently called LOS program, its many local variables have all been made global. To assure unique names for these global variables, and to aid in identifying them, each such variable name has been suffixed with .LS, so that TMACA in our discussions will appear as TMACA.LS in the computer code.

The mathematics of the LOS routine depends crucially on the following observation. Consider the elevation of a single hill I given by

$$\begin{aligned} Z(X,Y) &= \text{PEAK.H}(I) - \text{HT.H}(I) + \text{HT.H}(I) * \exp [Q_I(X,Y)] \\ &= C1 + C2 * \exp [Q_I(X,Y)] \end{aligned} \quad (20)$$

where Q_I is a negative definite quadratic function of X and Y in 2-space. Parameterize the straight line in X,Y space between an observer A and a target B as follows:

$$\begin{aligned} X(S) &= XA + S*(XB - XA) \\ Y(S) &= YA + S*(YB - YA) \end{aligned} \quad (21)$$

so that when $S=0$ we are at A 's location, and when $S=1$ we are at B 's location. Then consider the elevation of hill I as a function of S along the A to B line. The observation is that the elevation is given by a function of the same form as (20) except now with a single variable S .

$$Z(S) = C1 + C2 \exp [P(S)] \quad (22)$$

where $P(S)$ is a negative definite quadratic function of the parameter S .

To obtain the coefficients of $P(S)$ we plug $X(S)$ and $Y(S)$ from (21) into (20). For notational simplicity let XC, YC, PXX, PYY, PXY denote the computational parameters for hill I, then we have

$$\begin{aligned} Q_I(X,Y) &= Q_I(X(S),Y(S)) = \\ &= PXX*(X(S)-XC)^2 + PYY*(Y(S)-YC)^2 + PXY*(X(S)-XC)*(Y(S)-YC) \\ &= PXX*(XA+S*XBA-XC)^2 + PYY*(YA+S*YBA-YC)^2 \\ &\quad + PXY*(XA+S*XBA-XC)*(YA+S*YBA-YC) \end{aligned}$$

(where $XBA = XB - XA$ and $YBA = YB - YA$)

$$\begin{aligned} &= PXX*(RX+S*XBA)^2 + PYY*(RY+S*YBA)^2 \\ &\quad + PXY*(RX+S*XBA)*(RY+S*YBA) \end{aligned}$$

(where $RX = XA - XC$ and $RY = YA - YC$)

$$\begin{aligned} &= S^2*[PXX * XBA^2 + PYY * YBA^2 + PXY * XBA * YBA] \\ &\quad + S*[2PXX * XBA * RX + 2PYY * YBA * RY \\ &\quad + PXY * (XBA * RY + YBA * RX)] \\ &\quad + [PXX * RX^2 + PYY * RY^2 + PXY * RX * RY] \end{aligned}$$

$$= S^2 * GQ + S * FQ + EQ \quad \text{defining the quadratic coefficients}$$

GQ, FQ, EQ and giving the formula for $P(S)$. (23)

Given this equation, all future LOS computations can deal with the single dimensional computation along the A to B line parameterized by S .

We begin the discussion with a simple subroutine which is called from LOS in several places. Routine KOVER abstracts the LOS percent visible computation to its bare essence. Consider parameterizing an observer-to-target line in XY space with a distance parameter S so that $S=0$ represents the observer's position and $S=1$ represents the target's location. Assume:
(See Figure 18)

1. The observation device has elevation Z_0 .
2. The target has a macro-terrain elevation T_{MACT} , a micro terrain offset T_{MCT} and a size of $SIZET$. Thus its top is at $ZT = T_{MACT} + T_{MCT} + SIZET$.
3. At some point S between observer and target ($0 \leq S \leq 1$) there is an obstruction to line of sight whose top has elevation HTS .
4. Other previous computations have established a visible fraction of $VISFIN$ due to other obstructions.

To determine: Does the obstruction at S reduce the percent visible?

Let $ZS = Z_0 + S*(ZT - Z_0)$ (24)

If $HTS \geq ZS$, then the obstacle totally blanks LOS and the resulting percent visible is $VISFOUT=0$.

If $HTS < ZS$,

Let $EVIST = \max (T_{MACT}, Z_0 + (HTS - Z_0)/S)$ (25)

giving the elevation of the lowest point on the target which can be seen.

if $EVIST \geq ZT$ then LOS is blocked, and otherwise

$$VISFOUT = \min (VISFIN, (ZT-EVIST)/SIZET) \quad (26)$$

possibly decreasing the previously computed visible fraction $VISFIN$. Given the above formulas, the code for routine $KOVER$ should be self-explanatory. All variables in the routine are local. $KOVER$ is only called from LOS or from other routines which LOS calls. For the $KOVER$ program see Listing 3.

We now consider the various segments of the LOS routine in Listing 4.

A. Initialize

In addition to initializing $VISFRA$ and $VISFRB$, this segment of the code computes some temporary values which are used repeatedly later in the code. $KTREP$ is a global counter which is initialized to $-INF.C$ in $RES.TERR$ and increases by one on each call to LOS . It is used to avoid processing a


```

1 ROUTINE KOVER, GIVEN A1, I, NACT, SIZE1, Z1, S, HTS, Z5, VISTIN, YIELDING, VISEOUT
2 DEFINE Z0, I, NACT, SIZE1, Z1, S, HTS, Z5, VISEIN, VISEOUT, EVIST AS REAL VARIABLES
3 LET VISEOUT = VISEIN
4 IF S DE 0
5   IF HTS GT Z5 GO TO BLOCKED ELSE
6     LET EVIST = MAX, F(I, NACT, Z0 + (HTS - Z0) / S)
7     IF EVIST GE Z1 GO TO BLOCKED ELSE
8       IF EVIST LE Z1 - SIZE1 RETURN ELSE
9         LET VISEOUT = MIN, F(I, VISEOUT, (Z1 - EVIST) / SIZE1) RETURN
10      ELSE IF EVIST LT Z1 RETURN ELSE
11        BLOCKED, LET VISEOUT = 0. RETURN END

```

KOVER
Listing 3

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

given hill or forest ellipse more than one time. Note in lines 9 and 10 that if the micro terrain offset is negative, then part of the target is assumed masked by the terrain, so the visible fraction starts out at less than 1.0 .

B. Grid Square List

The grid square list is developed starting at grid square IX,IY which contains point B (lines 29-34). This grid square goes on the list as IGX(1), IGY(1). ISGX, ISGY are set to ± 1 depending on whether the grid square indices IX, IY increase or decrease as we move from B to A. (Lines 20-27). XSTEP and YSTEP are initialized (in lines 35-36) to the fraction of the distances XBA and YBA we have moved as we map out the grid squares from B to A . XINC and YINC are set to the increment in XSTEP and YSTEP which is added when we move across one grid square in the X or Y directions (Lines 20-27).

The loop from lines 37 to 47 sets XSTEP and YSTEP to the next X-grid intersection and the next Y-grid intersection. The smaller of XSTEP and YSTEP signals which intersection we hit first and hence which of IX, IY gets incremented by ISGX or ISGY. After each increment a new grid square is recorded in IGX, IGY. When both XSTEP and YSTEP are greater than 1.0 we have passed A, and the loop terminates with NGRSQ grid square indices recorded in the arrays IGX, IGY.

C. Forest Ellipse List and D. LOS Test at Forest Boundaries

If there are no forest ellipses on the battlefield (NCVELS=0) then this segment is bypassed. (Line 52). Otherwise we loop over all grid squares crossed, and for each grid square loop over all forest ellipses, IC, which intersect the grid square (lines 53-58). Lines 59-60 ensure that each ellipse is accessed at most one time.

For each such ellipse, we want to know whether or not the A to B line intersects the ellipse, and if so, where. The ellipse boundary equation is stored as

$$PXX*(X-XC)^2 + PYY*(Y-YC)^2 + PXY*(X-XC)*(Y-YC) = 1 \quad (27)$$

and X,Y on the A to B line are given parametrically as in (21).

Solving (21) and (27) simultaneously for S gives the intersection points (if any) as follows:

$$\begin{aligned} &PXX*(XA-XC + S*XBA)^2 + PYY*(YA-YC + S*YBA)^2 + \\ &PXY*(XA-XC + S*XBA)*(YA-YC + S*YBA) = 1 \\ &PXX*(RX + S*XBA)^2 + PYY*(RY + S*YBA)^2 + \\ &PXY*(RX + S*XBA)*(RY + S*YBA) = 1 \end{aligned} \quad (28)$$

where $RX = XA - XC$ and $RY = YA - YC$.

This simplifies as in (23) to the simple quadratic equation in S;

$$\begin{aligned} AA &= PXX*XBA^2 + PYY*YBA^2 + PXY*XBA*YBA \\ BB &= 2PXX*XBA*RX + 2PYY*YBA*RY + PXY*(XBA*RY + YBA*RX) \\ CC &= PXX*RX^2 + PYY*RY^2 + PXY*RX*RY - 1.0 \end{aligned} \quad (30)$$

Applying the quadratic formula gives the intersection points,

$$S1 = \frac{-BB - \sqrt{BB^2 - 4*AA*CC}}{2*AA} \quad (31)$$

$$S2 = \frac{-BB + \sqrt{BB^2 - 4*AA*CC}}{2*AA} \quad (32)$$

(with $S1 \leq S2$) if the radical is positive, and no intersection if the radical is negative. Lines 61-71 perform this computation.

If S1 and S2 exist then we need to consider whether the tree height at the forest boundaries S1, S2 interrupts LOS. Depending on whether one or both of S1 and S2 lie between A and B (that is between S=0 and S=1), and depending on whether A and B are air or ground platforms, it may be necessary to compute macro terrain elevation plus tree height at

either or both of S1 and S2 and see if the resulting obstacle decreases the percent visible. Subroutine TREE.CHECK does the terrain computation and LOS check (using KOVER), and will be documented following the LOS code. Lines (72-94) enumerate the possible cases which determine if TREE.CHECK is to be called at S1 and/or S2.

Finally, for this segment, the ellipse number IC, and the S1, S2 values for any ellipse which intersects the A-to-B line are saved in arrays IEL, CS1, CS2 for later use in the LOS procedure. (Lines 95-98).

E. Hilltops

Next the macro-terrain hills lying between A and B must be checked to see if they interrupt LOS. As indicated in Section V, we loop over all NGRSQ grid squares and for each, consult the LIST.H array to get the hill numbers I to be considered. (Lines 105-111). As for the forest ellipses, if a hill has already been considered in another grid square, then it is skipped over (Lines 112-113).

The only interesting computation in this section is finding the location W and height HHW of the hilltop. In general W will not be at the hill center, since the A to B line need not pass through the center. Thus HHW is usually somewhat less than PEAK.H(I). The hilltop location is easy to compute because of equations (22) and (23) which give the hill's elevation along the A to B line as

$$Z(S) = \text{PEAK.H}(I) - \text{HT.H}(I) + \text{HT.H}(I) * \exp[GQ * S^2 + FQ * S + EQ] \quad (33)$$

The hilltop must occur where $dZ/dS = 0$, so differentiating (33) and setting the result to zero gives

$$\frac{dZ}{dS} = \text{HT.H}(I) * \exp[GQ * S^2 + FQ * S + EQ] * [2GQ * S + FQ] = 0 \quad (34)$$

if and only if

$$[2GQ * S + FQ] = 0 \quad (35)$$

or

$$W = S = - FQ/(2*GQ) \quad (36)$$

Lines 114-124 compute GQ, FQ, EQ as in (23) and W as in (36). The hill height HHW at W is obtained by substituting W back into (33) giving

$$HHW=Z(W)=PEAK.H(I) - HT.H(I) + HT.H(I)*\exp[EQ-FQ^2/(4*GQ)] \quad (37)$$

in (lines 125-128). The tests of Figure 16 are performed in lines 129-141, and in particular, lines 133-139 loop through the previously saved forest ellipses to compute the tree height (if any) at W.

F. Lowest Sight Line

The lowest sight line procedure to compute the tangency point V, is an iterative procedure using the Newton-Raphson equation solving procedure. This is necessary because the equations describing the lowest sight line do not have a closed form solution. In the LOS code this computation is delegated to subroutine NEWTON (to be described shortly). The NEWTON procedure uses slightly different parameters depending on the direction of the computation A to B or B to A. Both calls are set up in lines 144-153 of the LOS code. The details of the NEWTON iteration will be documented along with routine NEWTON.

Finally, the LOS routine returns to the calling program either with the computed visible fractions (on line 157) after all hills have been checked in all NGRSQ grid squares, or with VISFRA = VISFRB = 0.0 if no LOS exists (line 159). This completes the detailed documentation of the LOS routine.

G. Routine TREE.CHECK

Given element A at S=0, element B at S=1, and a forest boundary at a point SS=S1 or S2 with $0 \leq SS \leq 1$, the TREE.CHECK routine tests whether the forest edge interrupts LOS from A to B or from B to A. The routine has no local variables, sharing the LS global variables with LOS.

```

1 ROUTINE LLS
2 DEFINE I,X,Y,U,L,IC,M
3 ** ALL VARIABLES EXCEPT THOSE DECLARED ABOVE ARE GLOBAL FOR USE IN LOS AND
4 ** ITS ASSOCIATED ROUTINES
5 LET VICFRA.LS = 1.0
6 LET XIA.LS = XA.LS - YA.LS
7 LET XBA.LS = XA.LS + YA.LS
8 IF XBA.LS EQ 0 AND YBA.LS LE 0.1 OF (SIZEF.LS + TMICB.LS LE 0.1) GO TO NO.LOS ELSE
9 IF XIA.LS + TMICA.LS LE 0.1 OF (SIZEF.LS + TMICB.LS LE 0.1) GO TO NO.LOS ELSE
10 IF TMICB.LS LE 0.1 OF (SIZEF.LS + TMICB.LS LE 0.1) GO TO NO.LOS ELSE
11 IF TMICB.LS LE 0.1 OF (SIZEF.LS + TMICB.LS LE 0.1) GO TO NO.LOS ELSE
12 LET ZA.LS = TMICA.LS + TMICB.LS + SIZEF.LS
13 LET ZB.LS = TMICA.LS + TMICB.LS + SIZEF.LS
14 LET ZC.LS = TMICA.LS + TMICB.LS + SIZEF.LS
15 LET XHSA.LS = XBA.LS ** 2 LET YHSA.LS = YBA.LS ** 2
16 LET XHBA.LS = XBA.LS ** 2 LET YHBA.LS = YBA.LS ** 2
17 LET LAG.LS = LAG.LS + LAG.LS LET CHTMAX.LS = 0.
18 ** COMPUTE LIST OF CRISOLAPES CROSSED BY A TO R LINE
19 LET NGFSG.LS = 0
20 IF XBA.LS LE 0.1 LET XBA.LS = 0.1 ALWAYS
21 IF XBA.LS GT 0.1 LET XBA.LS = 0.1 ALWAYS
22 LET ISGX.LS = -1 LET XINC.LS = GSIZE/XBA.LS JUMP AHEAD
23 LET ISGA.LS = 1 LET XINC.LS = -GSIZE/XBA.LS
24 HERE IF YBA.LS LE 0.1 LET YBA.LS = 0.1 ALWAYS
25 IF YBA.LS GT 0.1 LET YBA.LS = 0.1 ALWAYS
26 LET ISGY.LS = -1 LET YINC.LS = GSIZE/YBA.LS JUMP AHEAD
27 LET ISGL.LS = 1 LET YINC.LS = -GSIZE/YBA.LS
28 HERE IX.LS = 1 + TRUNC.F(XHSA.LS - X.LD.HORY)/GSIZE
29 LET IX.LS = 1 + TRUNC.F(XHBA.LS - Y.LD.HORY)/GSIZE
30 IF IX.LS LE 1 LET IX.LS = 1 ALWAYS
31 IF IX.LS GT 1 LET IX.LS = 1 ALWAYS
32 IF IX.LS GT 1 LET IX.LS = 1 ALWAYS
33 IF IX.LS GT 1 LET IX.LS = 1 ALWAYS
34 IF IX.LS GT 1 LET IX.LS = 1 ALWAYS
35 LET ASTEP.LS = (XBA.LS - X.LD.HORY - GSIZE*(IX.LS + 0.5*(ISGA.LS - 1.0)))/XBA.LS
36 LET YSTEP.LS = (YBA.LS - Y.LD.HORY - GSIZE*(IY.LS + 0.5*(ISGY.LS - 1.0)))/YBA.LS
37 ** GRADIENT **
38 LET ICGX.LS = (XBA.LS - X.LD.HORY) / XBA.LS LET ICGY.LS = (YBA.LS - Y.LD.HORY) / YBA.LS
39 IF ASTEP.LS LE 1.0 LET YSTEP.LS = 1.0
40 IF YSTEP.LS LE 1.0 LET XSTEP.LS = 1.0
41 ADD XINC.LS TO XSTEP.LS
42 ADD YINC.LS TO YSTEP.LS
43 ** STEP **
44 IF XSTEP.LS GT YSTEP.LS JUMP AHEAD
45 ELSE ADD ISGA.LS TO IX.LS
46 HERE ADD ISGY.LS TO IY.LS
47 GO TO GRIP.LOOP
48 ALWAYS ** GRIP LIST NOW COMPLETE IN IGX.LS, IGY.LS WITH NGFSG.LS LINES

```

LOS
Listing 4, Part 1

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDO


```

105 'SAVE ELL' IF LATOR.LS EQ 1 AND VISFRM.LS LE 0, GO TO Y100.LHS ELSE
106 IF LATOR.LS EQ 1 AND VISFRM.LS LE 0, GO TO Y100.LHS ELSE
107 ADD 1 TO NELS.LS
108 LET TEL.LS(NELS.LS) = TC
109 LET CS1.LS(NELS.LS) = S1.LS
110 IF CPK.LS GT CHTMAX.LS LET CHTMAX.LS = CPK.LS
111 IF 'PACK FOR NEXT GRID SQUARE' IN THIS GRID SQUARE
112 LOOP 'HACK FOR NEXT GRID SQUARE
113 'ALL ELLIPSES CHECKED AND SAVED
114 'HILL PROCESSING'
115 FOR K = 1 TO NGRKSU.LS DO
116 LET IX.LS = IGA.LS(K)
117 LET OX.LS(IX.LS) = LIST.HIX.LS,IX.LS,*)
118 LET N = DIM.FIDUM.I(IX.LS)
119 LET FASE.LS = DIM.I(IX.LS)
120 FOR
121 LET I = COM.I(IX.LS)
122 IF KNEP(IX.LS) EQ KTEP
123 LET KNEFF(IX.LS) = KTEP
124 ' COMPUTE W = TOP OF HILL'
125 LET PXX.LS = PXX.H(IX.LS)
126 LET KX.LS = XA.LS - XC.H(IX.LS)
127 LET GX.LS = PXX.LS*XA.HSQ.LS + PXX.LS*YHSAQ.LS + PXX.LS*YHSA.LS
128 LET FU.LS = 2*GX.LS*PXX.LS*RX.LS*YHSA.LS + PXX.LS*RX.LS*YHSA.LS +
129 PXX.LS*(IX.LS*YHSA.LS + RX.LS*YHSA.LS)
130 IF GX.LS EQ 0,0 CYCLE ELSE
131 LET W.LS = -FU.LS / (2*GX.LS)
132 IF ABSO.F(W.LS) GT 5,
133 LET FSO.LS = PXX.LS*PXX.LS*2
134 LET FU.LS = PXX.LS - FSO.LS
135 LET POW.LS = LT - 4,0 CYCLE ELSE
136 LET PK.LS = PAK.H(IX.LS)
137 LET HHW.LS = PK.LS + HT.LS*(EXP.F(FOW.LS)-1)
138 IF HHW.LS LE BASE.LS CYCLE ELSE
139 LET ZH.LS = ZA.LS + W.LS*ZRA.LS
140 IF ZH.LS LT 0, OR ZH.LS GT 1, JUMP AHEAD ELSE
141 IF HHW.LS GT ZH.LS GO TO Y100.LHS
142 IF HELL.LS EQ 0 JUMP AHEAD ELSE
143 LET CVHTM.LS = 0
144 IF W = 1 TO LELS.LS DO
145 IF (S1.LS(K) GT W.LS OR CS2.LS(K) LE W.LS CYCLE ELSE
146 LET IC = INT.LS(K)
147 IF (CVHTM.LS LT HT.F(IC) LET CVHTM.LS = HT.F(IC) ALWAYS
148 LOOP
149 IF DIM.LS + CVHTM.LS GE ZH.LS GO TO Y100.LHS
150 IF DIM.LS + CVHTM.LS LT HT.F(A.LS-SIZEA.LS,ZH.LS-SIZEA.LS) CYCLE ELSE

```

LOS

Listing 4, Part 3

```

142 ** IF WE GET TO HERE, THEN NEED TO FIND LOWEST SIGHT LINE OVER HILL
143 ** NEWTON ITERATION FROM A TO B GIVING VISFRA,LS
144 IF IATOB,LS EQ 1
145 LET Z7,LS = ZA,LS + HT,LS - PK,LS LET VSUB,LS = 0.
146 CALL NEWTON
147 IF VISFRA,LS LE 0. GO TO AC,LOS ELSE
148 ALWAYS
149 ** NEWTON ITERATION FROM P TO A GIVING VISFRA,LS
150 IF LBTHA,LS EQ 1
151 LET Z7,LS = ZB,LS + HT,LS - PK,LS LET VSUB,LS = 1.
152 CALL NEWTON
153 IF VISFRA,LS LE 0. GO TO AC,LOS ELSE
154 ALWAYS
155 LOOP ** BACK FOR NEXT HILL
156 ** BACK FOR NEXT GRID SQUARE
157 RETURN
158 ** NO,LOS, LET VISFRA,LS = 0. LET VISFRA,LS = 0.
159 RETURN END

```

LOS
Listing 4, Part 4

The TREE.CHECK Code is given in Listing 5

Line 2 computes the X and Y coordinates of the forest boundary at SS using equation (21).

Line 3 calls ELEV to get the macro terrain elevation HTS at SS

Line 4 adds to HTS the tree height for this forest ellipse

Lines 6-12 set up the A to B and B to A calls to routine KOVER to see if percent visible is decreased by the forest obstacle.

H. Routine NEWTON

Given the equation (33) for a hill parameterized in S, and given an observer's location at S=0 with observation device at elevation ZA, we wish to compute the point S=V at which the lowest sight line is tangent to the hill (if such a point exists), (See Figure 17). The elevation of the sight line at any S is given by the equation

$$L(S) = ZA + S*SLOPE \quad (38)$$

While the hill elevation is, as before,

$$Z(S) = PEAK.H-HT.H + HT.H*\exp \underbrace{[GQ*S^2 + FQ * S + EQ]}_{P(S)} \quad (39)$$

Tangency at V requires that the elevations be equal,

$$L(V) = Z(V) \quad (40)$$

and also that the slopes be equal at V .

$$L'(V) = Z'(V) \quad (41)$$

Unfortunately this set of equations does not have any closed form solution known to this author. Instead we apply the Newton Raphson iteration to solve for V in the following fashion:

```

1 ROUTINE TREE,CHECK
2 LET XS.LS = XA.LS + SS.LS*XA.LS
3 CALL ELEV GIVEN AS.LS, YS.LS YIELDING HTS.LS
4 ADD CPK.LS TO HTS.LS
5 LET ZS.LS = ZA.LS + SS.LS * ZRA.LS
6 IF LATOR.LS EQ 1
7   CALL KOVER(ZA.LS,THACE.LS,SIZE.LS,ZH.LS,SS.LS,HTS.LS,ZS.LS,VISF&R.LS)
8   YIELDING VISF&R.LS
9   ALWAYS
10 IF LHTUA.LS EQ 1
11   CALL KOVER(ZS.LS,THACE.LS,SIZE.LS,ZA.LS,1.0-SS.LS,HTS.LS,ZS.LS,VISF&A.LS)
12   YIELDING VISF&A.LS
13   ALWAYS
14   FUTURE
15   END

```

TREE,CHECK
Listing 5

THIS PAGE IS BEST QUALITY PRINTING
FROM COPY FURNISHED TO DDC

$$Z(V) = L(V) = ZA + L'(V)*V \quad (42)$$

by (40) and since $L(S)$ is linear. Thus, using (41),

$$Z(V) = ZA + Z'(V)*V \quad (43)$$

Also, from (34)

$$Z'(V) = HT.H*exp[P(V)]*P'(V) \quad (44)$$

Plugging (39) and (44) into (43) gives an equation for V :

$$\begin{aligned} \text{PEAK.H} - \text{HT.H} + \text{HT.H*exp}[GQ*V^2 + FQ*V + EQ] = \\ ZA + V*HT.H*exp[GQ*V^2 + FQ*V + EQ]*[2GQ*V+FQ] \end{aligned} \quad (45)$$

Or, collecting terms

$$\begin{aligned} \text{FCN}(V) \triangleq \text{HT.H*exp}[GQ*V^2+FQ*V+EQ]*(2GQ*V^2+F*V-1) \\ + ZA + \text{HT.H} - \text{PEAK.H} = 0 \end{aligned} \quad (46)$$

To find the V which makes $\text{FCN}(V) = 0$, we use the iterative Newton Raphson procedure, starting with the reasonable guess $V_1 = W$ and repeatedly applying the iteration formula

$$V_{k+1} = V_k - \frac{\text{FCN}(V_k)}{\text{DFCN}(V_k)} \quad (47)$$

where DFCN is the derivative of FCN . If the iteration leads to a V value which is remote from the location of A at $S=0$ and B at $S=1$, then the iteration is abandoned, and we cycle to the next hill. If the iteration converges to a V for which

$$Z(V) \approx ZA + Z'(V)*V \quad (48)$$

then this V is tested for its effect on LOS by calling the KOVER routine.

The actual code which follows in Listing 6 is slightly more involved than this description since it must be able to compute LOS in both directions. This is accomplished using the parameter VSUB which is 0 for an A to B call and 1 for a B to A call.

```

1 ROUTINE NEWTON
2 DEFINE M, IC AS INTEGER VARIABLES
3 ** ALL VARIABLES ARE REAL AND GLOBAL EXCEPT M, IC AS ABOVE AND
4 ** NCT, LS, NELS, LS, IEL, LS WHICH ARE INTEGER GLOBAL
5 LET NCT.LS = 0
6 LET V.LS = W.LS
7 LET HHV.LS = HHV.LS + HT.LS - PK.LS
8 LET TWOGV.LS = 2*GO.LS*V.LS
9 *TUP LET FCNV.LS = Z2.LS + HHV.LS*((FQ.LS+TWOGV.LS)*VM.LS-1.)
10 IF ABS.F(FCNV.LS) = HHV.LS*VM.LS*(TWOGV.LS**2 + 2*(GO.LS + TWOGV.LS*FQ.LS)+FSQ.LS)
11 IF ABS.F(FCNV.LS) LT C.CCCCCCCCCC RETURN ELSE
12 LET V.LS = V.LS - FCNV.LS/FCNV.LS
13 IF ABS.F(V.LS) GT 5. RETURN ELSE
14 LET VM.LS = V.LS - VSUR.LS
15 LET TWOGV.LS = 2*GO.LS*V.LS
16 LET POW.LS = FQ.LS + FQ.LS*V.LS + GO.LS*V.LS**2
17 IF POW.LS LT -4. RETURN ELSE
18 LET HHV.LS = HT.LS*EXP.F(PCNV.LS)
19 LET ELV.LS = Z2.LS + VM.LS*(HHV.LS*(FQ.LS+TWOGV.LS))
20 IF ABS.F(ELV.LS - HHV.LS) GT 1.
21 LET NCT.LS = NCT.LS + 1
22 IF NCT.LS LT 10 GO TO TUP ELSE
23 ALWAYS
24 IF V.LS LT 0. OR V.LS GT 1. RETURN ELSE
25 ** WE HAVE A GOOD VALUE OF V -- CHECK IT FOR FOREST COVERAGE
26 LET CVHTV.LS = C
27 FOR M = 1 TO NELS.LS DO
28 IF CSI.LS(M) GE V.LS CP CS2.LS(M) IF V.LS CYCLE ELSE
29 LET IC = IEL.LS(M)
30 IF CVHTV.LS LT HT.F(IC) LET CVHTV.LS = HT.F(IC) ALWAYS
31 LOOP
32 LET HTV.LS = HHV.LS + PK.LS + CVHTV.LS - HT.LS
33 LET ZV.LS = Z4.LS + V.LS*ZPA.LS
34 IF VSUB.LS EQ 0
35 CALL KOVER(ZV.LS, TMACH.LS, SIZEB.LS, ZB.LS, V.LS, HTV.LS, ZV.LS, VISFRB.LS) YIELDING
36 VISFR.LS
37 ELSE
38 CALL KOVER(ZV.LS, TMACH.LS, SIZEA.LS, ZA.LS, -VM.LS, HTV.LS, ZV.LS, VISFRA.LS) YIELDING
39 VISFRA.LS
40 ALWAYS RETURN END

```

NEWTON

Listing 6

I. Interface of LOS with STAR

To use the LOS routine the STAR model must set 14 parameters:

(all global variables with the .LS suffix)

XA, YA, TMACA, TMICA, SIZEA, LAGA, LATOB

XB, YB, TMACB, TMICB, SIZEB, LAGB, LBTOA

and CALL LOS. On return VISFRA and VISFRB have the fraction of A visible to B and the fraction of B visible to A respectively. If a one-way call was requested (say LBTOA = 0) then one of the visible fractions (VISFRA in this case) will contain a meaningless value and should be ignored.

Since the most frequent use for LOS is between entities in the simulation, a driver routine SIGHT has been written. (see Listing 7). Given the pointers A and B to the two entities, SIGHT sets up the 14 global variables from the appropriate entity attributes, and calls LOS. Note that this routine assumes both A and B are ground elements. The generalization to air elements is trivial. On return, the global variable PCT.VIS is set to VISFRB scaled by the size of the target B relative to the size of the M60 tank because the detection model in STAR is based on the DYN TACS field experiments which used the M60 as a target.

VII. Terrain Preprocessors and Data Input

The efficient computation of the STAR macro-terrain and forest representations requires prior execution of the HILL.LIST and TREE.LIST programs and the routine RES.TERR which reads terrain and forest data into the simulation.

A. HILL.LIST

As indicated in Section III, the HILL.LIST program preprocesses the macro terrain hill data to improve execution efficiency. The preprocessing consists of the following steps.

```

1 ROUTINE SIGHT(A,B)
2 DEFINE A,B AS INTEGER VARIABLES  ' ALL OTHERS ARE GLOBAL
3 LET XA.LS=X.CURRENT(A)
4 LET YA.LS=Y.CURRENT(A)
5 LET TMACA.LS=Z.CURRENT(A)
6 LET TMICA.LS=MICRO(A)
7 LET SIZEA.LS=TARDIM(SYS.TYPE(A),WPN.TYPE(A),4)
8 LET SIZEB.LS=TARDIM(SYS.TYPE(B),WPN.TYPE(B),4)
9 LET LAGA.LS=0
10 LET LATOB.LS=FWD.LOOK
11 CALL LOS
12 LET PCA.VIS(1)=VISFRA.LS
13 LET PCA.UNC(1)=1.0
14 LET PCT.VIS=VISFRA.LS*TARDIM(SYS.TYPE(B),WPN.TYPE(B),11)
15 RETURN
    END
    LET XB.LS=X.CURRENT(B)
    LET YB.LS=Y.CURRENT(B)
    LET TMACB.LS=Z.CURRENT(B)
    LET TMICB.LS=MICRO(B)
    LET LAGB.LS=0
    LET LBTOA.LS=BWD.LOOK
    LET PCB.VIS(1)=VISFRA.LS
    LET PCB.UNC(1)=1.0

```

SIGHT
Listing 7

1. Define the battlefield and an array of grid squares covering it.
2. Scan each grid square to build a list of which hills actually contribute to the macro terrain in that square.
3. For each grid square, sort the list of hills so that the biggest are first on the list. Thus if a hill interrupts LOS we are likely to find it sooner.
4. For each hill, find the lowest point on that hill that actually contributes to the macro terrain, and thus (possibly) improve the cutoff value for the hill.
5. Output a deck of data cards with hill parameters and the LIST.H array ready to be read into STAR by routine RES.TERR.

The Code is given in Listing 8.

Input to HILL.LIST is SIMSCRIPT free format from cards. The following values are required (in order): (Lines 5-6)

NGRIDX	}	number of grid squares in X and Y directions
NGRIDY		
GSIZE		size in meters of each grid square (eg. 1000.)
SPACING		increment in meters for scanning each grid square (eg. 50.)
X.LO.BDRY	}	battlefield coordinates in meters of the southwest corner origin of the grid square system
Y.LO.BDRY		
NHILLS		number of hills to be used on the battlefield.

```

1  ** MAIN PROGRAM HILL.LIST FOR PREPROCESSING MACRO TERRAIN HILL DATA TO
2  ** PRODUCE LIST.H ARRAY AND IMPROVED CUTOFFS FOR EACH HILL
3  **
4  ** PREAMBLE
5  ** MODE IS REAL
6  ** DEFINE NGRIDX, NGRIDY, NHILLS AS INTEGER VARIABLES
7  ** DEFINE IIX, IYY AS 1-DIMENSIONAL INTEGER ARRAYS
8  ** DEFINE BASE AS A 2-DIMENSIONAL INTEGER ARRAY
9  ** DEFINE LIST.H AS A 3-DIMENSIONAL INTEGER ARRAY
10 ** DEFINE TEMP, ZHILL AS 1-DIMENSIONAL INTEGER ARRAYS
11 ** DEFINE XC, YC, PEAK.H, HT.H, ANG.H, ECC.H, SPRD.H, CUT.H, XC.H, YC.H,
12 ** PXX.H, PYY.H, PXV.H, CRIT.H, MINZ.H AS 1-DIMENSIONAL REAL ARRAYS
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35

```

```

1  MAIN
2  NORMALLY MODE IS REAL
3  LET LINES.V = 70
4  DEFINE I.J.K.L.N.LMAX, IX.IV, JX, JY, NINCS, KOUNT AS INTEGER VARIABLES
5  READ NGRIDX, NGRIDY, GSIZE, SPACING, X.LO.BDRY, Y.LO.BDRY
6  READ NHILLS
7  RESERVE BASE(*,*) AS NGRIDX BY NGRIDY
8  RESERVE LIST.H(*,*,*) AS NGRIDX BY NGRIDY BY *
9  RESERVE TEMP(*); ZHILL(*) AS 100
10 RESERVE IIX(*), IYY(*) AS 25
11 RESERVE XC(*), YC(*), XC.H(*), YC.H(*), PEAK.H(*), HT.H(*), ECC.H(*),
12 SPRD.H(*), CUT.H(*), PXX.H(*), PYY.H(*), MINZ.H(*), CRIT.H(*), ANG.H(*),
13 DEFINE TITLE AS A 2-DIMENSIONAL ALPHA ARRAY
14 RESERVE TITLE(*,*) AS 3 BY NHILLS
15 FOR I = 1 TO NHILLS DO
16   START NEW CARD
17   READ TITLE(1,I), TITLE(2,I), TITLE(3,I), XC(I), YC(I), PEAK.H(I), ANG.H(I),
18   ECC.H(I), SPRD.H(I), HT.H(I) AS A 4, A 4, A 2, 7 D(10,0)
19   LET CUT.H(I) = HT.H(I)
20   LET A = LOG.E.F(HT.H(I)/(HT.H(I)-50.))
21   LET ANG = ANG.H(I)/RADIAN.C
22   LET PXX.H(I) = -(A*ANG**2+B*SANG**2)/(SPRD.H(I)**2)
23   LET PYY.H(I) = -(A*SANG**2+B*ANG**2)/(SPRD.H(I)**2)
24   LET PXV.H(I) = (2.*CANG*SANG*(B-A))/(SPRD.H(I)**2)
25   LET XC.H(I) = XC(I)*100.
26   IF CUT.H(I) GE HT.H(I) LET CRIT.H(I) = CUT.H(I)
27   ELSE LET CRIT.H(I) = HT.H(I)
28   ALWAYS LET MINZ.H(I) = PEAK.H(I)
29
30   LOOP
31   FOR IX = 1 TO NGRIDX DO
32   FOR IY = 1 TO NGRIDY DO
33   IF IX LT 10 LET BASE(IX,IY) = 230
34   ELSE IF IX LT 20 LET BASE(IX,IY) = 250
35   ELSE LET BASE(IX,IY) = 280
36   ALWAYS
37   LOOP

```

HILL.LIST

Listing 8, Part 1

```

36 LET NINCS = 1+INT.F(GSIZE/SPACING)
37 FOR IX = 1 TO NGFICX DO
38   FOR IY = 1 TO NGRIDY DO
39     LET KOUNT=0
40     LET X=X.LO.BDRY + GSIZE*(IX-1) LET Y=Y.LO.BDRY+GSIZE*(IY-1)
41     FOR JX = 1 TO NINCS DO
42       FOR JY = 1 TO NINCS DO
43         LET Z = BASE(IX,IY) CALL ELV1(X,Y,Z) YIELDING ZZ,I
44         LET Z = ZZ
45         IF I NE 0
46           IF Z LT MINZ.H(1) LET MINZ.H(1)=Z
47           ALWAYS
48           IF KOUNT NE 0
49             FOR N=1 TO KOUNT DO
50               IF I EQ TEMP(N) LET ZHILL(N)=ZHILL(N)+Z GO TO NEXTY
51             ELSE
52               LOOP
53               ALWAYS ADD 1 TO KOUNT
54               LET TEMP(KOUNT)=I
55               LET ZHILL(KOUNT)=Z
56               ALWAYS 'NEXTY' ADD SPACING TO Y
57               LOOP
58               LET Y = Y.LO.BDRY +GSIZE*(IY-1)
59               LOOP
60               IF KOUNT NE 0
61                 FOR K = 1 TO KOUNT DO
62                   LET ZMAX = -1000. LET LMAX = 0
63                   FOR L = K TO KOUNT DO
64                     IF ZHILL(L) GT ZMAX
65                       LET ZMAX=ZHILL(L) LET LMAX=L
66                   ALWAYS
67                   LOOP
68                   LET N = TEMP(K) LET ZI = ZHILL(K) LET TEMP(K)=TEMP(LMAX)
69                   LET ZHILL(K)=ZHILL(LMAX) LET TEMP(LMAX)=ZI
70                   LOOP
71                   ALWAYS
72                   RESERVE LIST.H(IX,IY,*) AS KOUNT+1
73                   LET LIST.H(IX,IY,1)=BASE(IX,IY)
74                   IF KOUNT NE 0
75                     FOR N = 1 TO KOUNT
76                       LET LIST.H(IX,IY,N+1)=TEMP(N)
77                     ALWAYS LOOP LOOP 'ALL GRID SQUARES NOW PROCESSED NOW OUTPUT
78

```

HILL.LIST

Listing 8, Part 2


```

106 START NEW PAGE
107 FOR IX = 1 TO NGRIDX DO
108   FOR IY = 1 TO NGRIDY DO
109     LET KOUNT = DIM.F(LIST.H(IX,IY,*))-1
110     START NEW OUTPUT LINE
111     PRINT 1 LINE WITH LINE
112     IY = ***** NO. HILLS = ***** BASE = *****
113     FOR I=2 TO KOUNT+1 WRITE LIST.H(IX,IY,I) AS (16)I 6
114     USE UNIT 7 FOR OUTPUT
115     PRINT 1 LINE WITH IX,IY,KOUNT,LIST.H(IX,IY,I) AS FOLLOWS
116     *****
117     FOR I = 2 TO KOUNT + 1 WRITE LIST.H(IX,IY,I) AS (16)I 5
118     USE UNIT 6 FOR OUTPUT
119     LOOP LOOP
120     NOW TALLY FOR EACH HILL
121     START NEW PAGE
122     FOR I = 1 TO NHILLS DO
123       LET J = 0
124       FOR IY = 1 TO NGRIDY DO
125         FOR IX = 1 TO NGRIDX DO
126           LET KOUNT = DIM.F(LIST.H(IX,IY,*))
127           FOR L = 2 TO KOUNT+1 DO
128             IF LIST.H(IX,IY,L) EQ 1
129               ADD 1 TO J
130               LET IIX(J) = IX
131               LET IIY(J) = IY
132             ALWAYS
133             LOOP LOOP LOOP
134             START NEW OUTPUT LINE
135             PRINT 1 LINE WITH I,J AS FOLLOWS
136             APPEARS IN ***** GRID SQUARES
137             FOR L=1 TO J WRITE IIX(L) AS (25)I 5
138             FOR L=1 TO J WRITE IIY(L) AS (25)I 5
139             LOOP
140             STOP
141             END
142
143 ROUTINE ELV(X,Y,Z) YIELDING ZZ,J
144 NORMALLY MODE IS REAL
145 DEFINE I,J AS INTEGER VARIABLES
146 LET J=0
147 LET ZZ = Z
148 FOR I = 1 TO NHILLS DO
149   LET XS = X - XC.H(I)
150   LET YS = Y - YC.H(I)
151   LET VS = XS + PXY.H(I)*XS*YS
152   LET FACTOR = PXX.H(I)*XS*XS + PYY.H(I)*YS*YS + PXY.H(I)*XS*YS
153   IF FACTOR LT CRIT.H(I) CYCLE
154   ELSE LET ELV = PEAK.H(I)+HT.H(I)*(EXP.F(FACTOR)-1.)
155   IF ELV GT ZZ LET ZZ = ELV
156   LET J = I
157 ALWAYS LOOP RETURN END

```

HILL.LIST
Listing 8, Part 4

Next the fitting parameters for each hill are input: (line 17-18)

XC.H, YC.H, PEAK.H, ANG.H, ECC.H, SPRD.H, HT.H.

The current program does not read CUT.H since cutoffs were not used in the original battlefield coding. These could easily be added. The fitting parameters are converted to computing parameters in lines 19-27.

Finally, BASE values for each grid square are entered. The current code has these hard wired since only 3 distinct base values were used on the original battlefield. A card read could easily replace this. (Lines 30-35).

Lines 37-59 perform the battlefield scan, computing elevation every SPACING meters and recording which hill was responsible for the final elevation. This hill number is entered in a temporary array TEMP if it is not already there. (Lines 48-55). Array ZHILL records the contribution of each hill to this grid square for the upcoming sort.

Lines 60-78 sort the TEMP array and store it in the appropriate column of LIST.H. The rest of the program prints and punches the output data deck. This deck is ready for immediate input to the simulation via routine RES.TERR .

B. TREE.LIST

in Listing 9

The TREE.LIST program has exactly the same structure as the HILL.LIST program. It is presented here with essentially no further comment. The data output deck from TREE.LIST is ready to be read by RES.TERR except that the first card (battlefield definition) which duplicates the first card from HILL.LIST should be discarded.

C. RES.TERR Routine

RES.TERR is a routine called from the STAR MAIN program which initializes the following data:


```

23 LET NIMCS = 1+INT.F(GSIZE/SPACING)
24 FOR IX = 1 TO NGRIDX DO
25   FOR IY = 1 TO NGRIDY DO
26     LET KOUNT=0
27     LET X=X.LO.BDRY + GSIZE*(IX-1) LET Y=Y.LO.BDRY+GSIZE*(IY-1)
28     FOR JX = 1 TO NIMCS DO
29       FOR JY = 1 TO NIMCS DO
30         FOR I = 1 TO NCVELS DO
31           LET XS = X -XC.E(I) LET YS = Y - YC.E(I)
32           LET FACTOR = PXX.E(I)*XS*XS + PYY.E(I)*YS*YS + PXY.E(I)*XS*YS
33           IF FACTOR LT 1
34             IF KOUNT NE 0
35               FOR N = 1 TO KOUNT DO
36                 IF I EQ TEMP(N) GO TO NEXTI
37             ELSE
38               LOOP
39               ALWAYS ADD 1 TO KOUNT
40               LET TEMP(KCUNT) = I
41             ALWAYS
42           •NEXTI• LOOP SPACING TO Y
43           ADD ADD SPACING TO X
44           LOOP LET Y = Y.LO.BDRY +GSIZE*(IY-1)
45           LOOP
46           •NOW SORT TEMP FOR THIS GRID SQUARE
47           IF KOUNT NE 0
48             FOR K = 1 TO KCOUNT DO
49               LET ZMAX = -1000. LET LMAX = 0
50               FOR L = K TO KOUNT DO
51                 IF HT.E(TEMP(L)) GT ZMAX
52                   LET ZMAX = HT.E(TEMP(L)) LET LMAX=L
53               ALWAYS
54             LOOP
55             LET N = TEMP(K)
56             LET TEMP(LMAX) = N
57             LOOP
58             ALWAYS
59             RESEKVE LIST.E(IX,IY,*) AS KOUNT + 1
60             LET LIST.E(IX,IY,1) = KCUNT
61             IF KOUNT NE 0
62               FOR N = 1 TO KCOUNT
63                 LET LIST.E(IX,IY,N+1) = TEMP(N)
64             ALWAYS
65             LOOP LOOP ••ALL GRID SQUARES NOW PROCESSED NOW OUTPUT
66

```

TREE.LIST

Listing 9, Part 2

```

67 START NEW PAGE
68 PRINT 1 LINE WITH NGRIDX,NGRIDY,GSIZE,SPACING AS FOLLOWS
69 NGRIDX = **** NGRIDY = **** GSIZE = **** SPACING = ****
70 PRINT 1 LINE WITH X.LO.BDRY,Y.LO.BDRY AS FOLLOWS
71 BOUNDARIES ARE X.LO.BDRY = **** Y.LO.BDRY = ****
72 USE UNIT 7 FOR OUTPUT
73 PRINT 1 LINE WITH NGRIDX,NGRIDY,GSIZE,X.LO.BDRY,Y.LO.BDRY AS FOLLOWS
74 PRINT 1 LINE WITH NCVELS AS FOLLOWS
75 USE UNIT 6 FOR OUTPUT
76 PRINT 1 DOUBLE LINE AS FOLLOWS
77 TITLE XC YC DO HT ANGLE AMAJ AMIN PXX PYY PXY
78 FOK I = 1 TO NCVELS DO
79 PRINT 1 DOUBLE LINE WITH I,TITLE(1,I),TITLE(2,I),TITLE(3,I),XC(I),YC(I),
80 HT.E(I),ANG.E(I),AMAJ.E(I),AMIN.E(I),PXX.E(I),PYY.E(I),PXY.E(I)
81 AS FOLLOWS
82 *****
83 *****
84 *****
85 *****
86 *****
87 *****
88 *****
89 *****
90 *****
91 *****
92 *****
93 *****
94 *****
95 *****
96 *****
97 *****
98 *****

```

TREE.LIST
Listing 9, Part 3

```

99      ** NUM TALLY FOR EACH ELLIPSE
100     START NEW PAGE
101     FOR I = 1 TO NCVELS DO
102       LET J = 0
103       FOR IX = 1 TO NGRIDX DO
104         FOR IY = 1 TO NGRIDY DO
105           LET KOUNT = LIST.E(IX,IY,1)
106           FOR L = 1 TO KOUNT DO
107             IF LIST.E(IX,IY,L+1) EQ 1
108               ADD 1 TO J
109               LET IIX(J) = IX
110               LET IIY(J) = IY
111             ALWAYS
112           LOOP LOOP LOOP
113         START NEW OUTPUT LINE
114         PRINT 1 LINE WITH I,J AS FOLLOWS
115     ELLIPSE *** APPEARS IN *** GRID SQUARES
116     FOR L = 1 TO J WRITE IIX(L) AS (25)I 5
117     FOR L = 1 TO J WRITE IIY(L) AS (25)I 5
118   LOOP
119 STOP
END

```

TREE.LIST
 Listing 9, Part 4

1. Battlefield definition and grid square system
2. Macro-terrain hill computing parameters
3. LIST.H array
4. Forest ellipse computing parameters
5. LIST.C array

Input to RES.TERR consists of

1. The punched output deck from HILL.LIST followed by
2. Either a single card with a 0 (zero) if there are no forest ellipses or else the punched output deck from TREE.LIST minus its first card (which duplicates the battlefield definition card from HILL.LIST)

RES.TERR is set up to dynamically reserve and dimension the various arrays so that no more core is required than is absolutely necessary. For example, if there are no forest ellipses, then no data arrays are reserved for ellipse parameters or for LIST.C. This first implementation of RES.TERR makes no attempt to pack data values, although if required some packing should be possible. The program listing appears in Listing 10.

Code:	Lines 6-12	Define the battlefield and dimension the hill arrays
	Lines 14-30	Read the hill fitting parameters and convert to computing parameters
	Lines 31-37	Reserve and read in the ragged array LIST.H
	Lines 38-39	Determine whether there are any forest ellipses
	Lines 40-42	Dimension the forest ellipse arrays
	Lines 43-54	Read the ellipse fitting parameters and convert to computing parameters
	Lines 55-61	Reserve and read in the ragged array LIST.C .

```

1 ROUTINE RES.TERR
2 **ROUTINE TO RESERVE AND READ IN DATA ARRAYS FOR TERRAIN HILLS, COVER
3 ** ELLIPSES, AND BATTLEFIELD COORDINATES
4 NORMALLY CODE IS 0000
5 DEFINE I,IA,IV,KOUNT,J,JA,JY
6 USE UNIT 14 FOR INPUT
7 READ MGRIOX,MGRIOY,GSIZE,X,LC,PDY,Y,LC,PDY,HILLS **ALL GLOBAL
8 RESERVE ICH,X,LS(*),ICX,LS(*),ICX,LS(*) AS NGRIDY
9 RESERVE ICH,X,LS(*),ICX,LS(*),ICX,LS(*) AS NGRIDY
10 RESERVE ICH,X,LS(*),ICX,LS(*),ICX,LS(*) AS NGRIDY BY *
11 RESERVE XCH,X,LS(*),ICX,LS(*),ICX,LS(*),ICX,LS(*),ICX,LS(*)
12 AS NGRIDY
13 RESERVE KREPER(*) AS NGRIDY
14 LET KREPER=1000
15 FOR I=1 TO NGRIDY
16   READ J
17   IF J=0 PRINT A LINE WITH I AS FOLLOWS
18   INPUT DATA SLU,FACE,ELV IN HILL DATA FOR HILL ***** XXXXX
19   ALWAYS
20   READ XCH,X,LS(*),ICX,LS(*),ICX,LS(*),ICX,LS(*)
21   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
22   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
23   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
24   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
25   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
26   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
27   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
28   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
29   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
30   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
31   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
32   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
33   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
34   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
35   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
36   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
37   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
38   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
39   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
40   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
41   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
42   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
43   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
44   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
45   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
46   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
47   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
48   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
49   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
50   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
51   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
52   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
53   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
54   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
55   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
56   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
57   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
58   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
59   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
60   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
61   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
62   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
63   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
64   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
65   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
66   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
67   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
68   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
69   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
70   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
71   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
72   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
73   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
74   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
75   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
76   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
77   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
78   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
79   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
80   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
81   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
82   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
83   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
84   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
85   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
86   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
87   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
88   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
89   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
90   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
91   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
92   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
93   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
94   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
95   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
96   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
97   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
98   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
99   LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)
100  LET XCH,X,LS(*),ICX,LS(*),ICX,LS(*)

```

RES.TERR

Listing 10, Part 1

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```

32 FOR IX = 1 TO NGRIDX DO
33 FOR IY = 1 TO NGRIDY DO
34 READ JX,JY,KOUNT
35 IF IX NE JX OR IY NE JY PRINT 1 LINE WITH IX,IY AS FOLLOWS
XXXXX INPUT DATA SEQUENCE CHECK IN LIST.H DATA FOR GRID ***** XXXXX
36 ALWAYS RESERVE LIST.H(IX,IY,*) AS KOUNT+1
37 FOR I = 1 TO KOUNT+1 READ LIST.H(IX,IY,I)
38 LOOP
39 READ NCVELS
40 IF NCVELS EQ 0 USE UNIT 5 FOR INPUT RETURN
41 ELSE RESERVE LIST.H(IX,IY,*) AS NGRIDX BY NGRIDY BY *
42 RESERVE AC(IX,IY,*) AS PXX.E(*),PT.E(*),PVV.E(*),PXY.E(*),KCREP(*) AS NCVELS
43 FOR I = 1 TO NCVELS DO
44 READ J
45 IF I EQ J PRINT 1 LINE WITH I AS FOLLOWS
XXXXX INPUT DATA SEQUENCE CHECK IN COVER ELLIPSE NUMBER ***** XXXXX
46 ALWAYS
47 READ AC,YC,MT,E(I),ANG,AVV,J,AMTH
48 LET ANG = ANG/RADIAT.C LET SANG = SIN.F(ANG) LET CANG = COS.F(ANG)
49 LET PXX.E(I) = (CANG/AMTH)**2 + (SANG/AMTH)**2
50 LET PVV.E(I) = (SANG/AMTH)**2 + (CANG/AMTH)**2
51 LET PXY.E(I) = 2.*SANG.CANG*(1./AMTH**2 - 1./AMTH**2)
52 LET AC.E(I) = XC + 100. LET YC.E(I) = YC + 100.
53 LET KCREP(I) = -1.E-6
54 LOOP
55 FOR IX = 1 TO NGRIDX DO
56 FOR IY = 1 TO NGRIDY DO
57 READ JX,JY,KOUNT
58 IF IX NE JX OR IY NE JY PRINT 1 LINE WITH IX,IY AS FOLLOWS
XXXXX INPUT DATA SEQUENCE CHECK IN LIST.H DATA FOR GRID ***** XXXXX
59 ALWAYS RESERVE LIST.H(IX,IY,*) AS KOUNT+1
60 LET LIST.H(IX,IY,I) = KOUNT
61 FOR I = 2 TO KOUNT + 1 READ LIST.H(IX,IY,I)
62 LOOP
63 USE UNIT 5 FOR INPUT
64 RETURN

```

RES.TERR

Listing 10, Part 2

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

VIII. Some Areas for Further Development

The current version of the STAR terrain and LOS models form a working core for terrain modelling in STAR. There are several areas in which work remains to be done. A preliminary list is presented here.

1. Incorporation of terrain features from the Army Mobility Model patch data for micro terrain, concealment, vehicle limiting speeds (for the movement model), etc.
2. Development of a dynamic smoke model in conjunction with detection models for the "dirty battlefield".
3. Investigation of computer assisted terrain fitting to speed the hill definition task and possibly result in better terrain fitting.
4. Replacement of the iterative Newton algorithm in LOS with a guaranteed accuracy approximation using splines or some other appropriate class of functions.

Work is currently underway on several of these areas. As they become ready for incorporation into the main STAR model, they will be documented separately.

REFERENCES

1. NEEDELS, C. J., Parameterization of Terrain in Army Combat Models,
M.S. Thesis, Naval Postgraduate School, Monterey, March 1976.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2. Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3. Department Chairman, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
4. Professor James K. Hartman Code 55Hh Department of Operations Research Naval Postgraduate School Monterey, California 93940	30
5. Professor S. H. Parry, Code 55Py Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
6. LTC Edward P. Kelleher, Code 55Ka Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
7. Professor Arthur L. Schoenstadt, Code 53Zh Department of Mathematics Naval Postgraduate School Monterey, California 93940	1
8. Office of the Commanding General U.S. Army TRADOC Attn: General Donn A. Starry Ft. Monroe, Virginia 23651	1
9. Headquarters U.S. Army Training & Doctrine Command Attn: ATCG-T (Col. Ed Scribner) Ft. Monroe, Virginia 23651	1

10. Headquarters
U.S. Army Training & Doctrine Command
Attn: Director, Analysis Directorate
Combat Developments (MAJ Chris Needels)
Ft. Monroe, Virginia 23651 1
11. Headquarters
U.S. Army Training & Doctrine Command
Attn: Director, Maneuver Directorate
Combat Developments (Col. Fred Franks)
Ft. Monroe, Virginia 23651 1
12. Mr. David Hardison
Deputy Under Secretary of the Army
(Operations Research)
Department of the Army, The Pentagon
Washington, D.C. 20310 1
13. Lt. General J. R. Thurman
Commanding General
U.S. Army Combined Arms Center
Ft. Leavenworth, Kansas 66027 1
14. Director
Combined Arms Combat Development Activity
Attn: Col. Reed
Ft. Leavenworth, Kansas 66027 1
15. Director, BSSD
Combined Arms Training Development Activity
Attn: ATZLCA-DS
Ft. Leavenworth, Kansas 66027 1
16. Director
Combat Analysis Office
Attn: Mr. Kent Pickett
U.S. Army Combined Arms Center
Ft. Leavenworth, Kansas 66027 1
17. Command and General Staff College
Attn: Education Advisor
Room 123, Bell Hall
Ft. Leavenworth, Kansas 66027 1
18. Dr. Wilbur Payne, Director
U.S. Army TRADOC Systems Analysis Activity
White Sands Missile Range, New Mexico 88002 1
19. Headquarters, Department of the Army
Office of the Deputy Chief of Staff
for Operations and Plans
Attn: DAMO-2D
Washington, D.C. 20310 1

20. Commander 1
U.S. Army Concepts Analysis Agency
8120 Woodmont Avenue
Attn: MOCA-SMS (CPT Steve Shupack)
Bethesda, Maryland 20014
21. Commander 1
U.S. Army Concepts Analysis Agency
Attn: LTC Earl Darden-MOCA-WG
8120 Woodmont Avenue
Bethesda, MD. 20014
22. Director 1
U.S. Army Night Vision & Electro-optical Lab.
Attn: DEL-NV-VI-Mr. Bob Hermes
Fort Belvoir, VA 22060
23. Director 1
U.S. Army Material Systems Analysis Activity
Attn: Mr. Will Brooks
Aberdeen Proving Grounds, Maryland 21005
24. Director 1
Combat Developments Experimentation Command
Attn: CPT William J. Caldwell
Fort Ord, California 93941
25. Director 1
Armored Combat Vehicle Technology Program
Attn: Col. Fitzmorris
U.S. Army Armor Center
Ft. Knox, Kentucky 40121
26. Col. Frank Day 1
TRADOC Systems Manager - XM1
U.S. Army Armor Center
Ft. Knox, Kentucky 40121
27. Director 1
Combat Developments, Studies Division
Attn: MAJ W. Scott Wallace
U.S. Army Armor Agency
Fort Knox, KY 40121
28. Commandant 1
U.S. Army Field Artillery School
Attn: ATSF-MBT (CPT Steve Starnier)
Fort Sill, Oklahoma 73503
29. Director 1
Combat Developments
Attn: Col. Clark Burnett
U.S. Army Aviation Agency
Fort Rucker, Alabama

30. Director 1
Combat Developments
U.S. Army Infantry Agency
Fort Benning, GA
31. Director 1
Missile Intelligence Agency
Attn: ADA Tactics (CPT E.G. Hagewood)
Redstone Arsenal, AL 35809
32. Director 1
Combat Developments
Attn: CPT William D. Meiers
U.S. Army Air Defense Agency
Fort Bliss, TX 79905
33. LTC J. E. Kim 1
PPBS
Dept. Ministry of National Defense
Republic of Korea
34. Commander 1
U.S. Army Logistics Center
ATTN: ATCL-OS-Mr. Cammeron/CPT Schuessler
Fort Lee, VA 23801
35. Commander, USAMMCS 1
ATTN: ATSK-CD-CS-Mr. Lee/Mr. Marmon
Redstone Arsenal, AL 35809
36. R. Stampfel, Code 55 1
Department of Operations Research
Naval Postgraduate School
Monterey, California 93940